

PlanetPress® Connect

OL™ Software

User Guide

Version: 1.8

PlanetPress® Connect

OL™ Software

User Guide

Version 1.8

Last Revision: 2018-04-09

Objectif Lune, Inc.

2030 Pie-IX, Suite 500

Montréal, QC, Canada, H1V 2C8

+1 (514) 875-5863

www.objectiflune.com

All trademarks displayed are the property of their respective owners.

© Objectif Lune, Inc. 1994-2018. All rights reserved. No part of this documentation may be reproduced, transmitted or distributed outside of Objectif Lune Inc. by any means whatsoever without the express written permission of Objectif Lune Inc. disclaims responsibility for any errors and omissions in this documentation and accepts no responsibility for damages arising from such inconsistencies or their further consequences of any kind. Objectif Lune Inc. reserves the right to alter the information contained in this documentation without notice.

Table of Contents

- Table of Contents** **4**
- Welcome to PlanetPress Connect 1.8** **14**
- Setup And Configuration** **15**
 - System and Hardware Considerations 15
 - Antivirus Exclusions 15
 - Database Considerations 17
 - Environment Considerations 22
 - Language and Encoding Considerations 24
 - Network Considerations 24
 - Performance Considerations 25
 - System Requirements 27
- Installation and Activation 28
 - Where to obtain the installers 28
 - Installation - important information 29
 - Installation - "How to" guides 29
 - Activation 29
 - Installation Prerequisites 29
 - User accounts and security 31
 - The Importance of User Credentials on Installing and Running PlanetPress Connect 31
 - Installing PlanetPress Connect on Machines without Internet Access 33
 - Installation Wizard 35
 - Running connect installer in Silent Mode 41
 - Activating a License 49
 - Migrating to a new workstation 52
 - Information about PlanetPress Workflow 8 59
 - Upgrading from PlanetPress Suite 6/7 60
 - What do I gain by upgrading to PlanetPress Connect? 63
- Known Issues 79
 - Issues with Microsoft Edge browser 79
 - Worklfow - "Execute Data Mapping" - Issues with mutliple PDFs 79
 - Installation Paths with Multi-Byte Characters 79
 - Switching Languages 79
 - GoDaddy Certificates 80

MySQL Compatibility	80
PostScript Print Presets	80
Available Printer Models	81
External Resources in Connect	81
Using Capture After Installing Workflow 8	82
Capturing Spool Files After Installing Workflow 8	82
Colour Model in Stylesheets	82
Image Preview in Designer	82
Merge\Weaver Engines when Printing	82
REST Calls for Remote Services	83
Print Content and Email Content in PlanetPress Workflow	83
Print Limitations when the Output Server is located on a different machine	83
VIPP Output	83
Server Configuration Settings	84
Scheduling Preferences	85
Server Security Settings	90
Uninstalling	91
Important Note: Stop any active Anti-Virus software before uninstalling Connect.	91
Impacts upon other Applications and Services	91
Uninstallation Wizard	92
General information	93
Connect: a peek under the hood	93
The Workflow server	94
The Connect server	95
The Connect database	96
The File Store	96
The engines	97
The REST API	97
Connect File Types	98
The DataMapper Module	100
DataMapper basics	100
What's next?	101
Data mapping configurations	101
Creating a new data mapping configuration	102
Opening a data mapping configuration	105
Saving a data mapping configuration	106

Using the wizard for CSV and Excel files	106
Using the wizard for databases	108
Using the wizard for PDF/VT and AFP files	111
Using the wizard for XML files	112
Data mapping workflow	113
Creating a data mapping workflow	113
Testing the extraction workflow	115
Data source settings	115
Extracting data	118
Steps	140
The Data Model	151
Creating a Data Model	152
Editing the Data Model	153
Using the Data Model	154
Fields	155
Detail tables	161
Data types	168
Data Model file structure	177
DataMapper User Interface	180
Keyboard shortcuts	181
Menus	186
Panels	190
Example	196
Settings for location-based fields in a Text file	221
Settings for location-based fields in a PDF File	221
Settings for location-based fields in CSV and Database files	222
Settings for location-based fields in an XML File	223
Text and PDF Files	226
CSV and Database Files	228
XML File	229
Text and PDF Files	233
CSV and Database Files	235
XML Files	237
Left operand, Right operand	241
Condition	243
Operators	243
Text file	244

PDF File	245
CSV File	247
XML File	247
JavaScript	249
Toolbar	249
Welcome Screen	251
DataMapper Scripts API	252
Using scripts in the DataMapper	255
Setting boundaries using JavaScript	257
Objects	262
Example	266
Example	268
Examples	271
Example	273
Example	274
Example	276
Examples	279
Examples	282
Example	285
Example	286
Example	290
Text	291
XML	292
Functions	292
The Designer	302
Designer basics	302
Features	303
Templates	304
Contexts	320
Sections	321
Print	325
Copy Fit	327
Creating a Print template with a Wizard	327
Print context	332
Print sections	335
Pages	343
Master Pages	350

Media	353
Email	359
Designing an Email template	361
Creating an Email template with a Wizard	364
Email context	368
Email templates	370
Email header settings	373
Email attachments	379
Web	381
Creating a Web template with a Wizard	382
Web Context	386
Web pages	387
Forms	393
Using Form elements	398
Using JavaScript	403
Capture OnTheGo	406
COTG Forms	406
Creating a COTG Form	407
Filling a COTG template	408
Testing the template	409
Sending the template to the Workflow tool	410
Using COTG data in a template	410
Designing a COTG Template	413
Capture OnTheGo template wizards	416
Using Foundation	420
COTG Elements	423
Using COTG Elements	431
Testing a Capture OnTheGo Template	436
Using the COTG plugin: cotg-2.0.0.js	442
Dynamically adding COTG widgets	445
Saving and restoring custom data and widgets	449
Capture OnTheGo API	453
Content elements	465
Element types	465
Editing HTML	466
Attributes	467
Inserting an element	468

Selecting an element	469
Deleting an element	469
Styling and formatting an element	470
Barcode	470
Boxes	513
Business graphics	516
COTG Elements	519
Date	526
Forms	527
Form Elements	532
Hyperlink and mailto link	536
Images	537
Table	542
Text and special characters	546
Snippets	548
Adding a snippet to the Resources	549
Adding a snippet to a section	550
Creating a snippet	550
JSON Snippets	551
Styling and formatting	551
Local formatting versus style sheets	551
Layout properties	552
Styling templates with CSS files	553
Styling text and paragraphs	562
How to position elements	567
Rotating elements	570
Styling a table	571
Styling an image	576
Background color and/or image	579
Border	580
Colors	583
Fonts	587
Locale	590
Spacing	591
Personalizing Content	592
Variable data	592
Conditional content	593

Dynamic images	593
Dynamic tables	593
Snippets	594
Scripts	594
Loading data	594
Variable Data	604
Formatting variable data	610
Showing content conditionally	613
Conditional Print sections	616
Dynamic Images	617
Dynamic table	618
Personalized URL	623
Writing your own scripts	624
Script types	624
Creating a new script	625
Writing a script	627
Managing scripts	629
Testing scripts	632
Optimizing scripts	636
Loading a snippet via a script	640
Loading content using a server's RESTful API	643
Control Scripts	645
The script flow: when scripts run	660
Selectors in Connect	660
Designer User Interface	666
Dialogs	666
Keyboard shortcuts	738
Menus	744
Panels	755
Toolbars	771
Welcome Screen	776
Print Options	777
Job Creation Presets	840
Output Creation Settings	850
Designer Script API	873
Designer Script API	874
Examples	882

Examples	883
Examples	884
Examples	886
Examples	889
Examples	889
Examples	890
Examples	892
Examples	892
Examples	894
Examples	895
Examples	895
Examples	896
Example	897
Example	897
Example	898
Example	898
Example	899
Examples	900
Creating a table of contents	900
Example	902
Examples	902
Examples	905
Examples	906
Examples	906
Replace elements with a snippet	907
Replace elements with a set of snippets	907
Example	908
Example	908
Creating a Date object from a string	917
Control Script API	930
Examples	944
Generating output	953
Print output	953
Email output	953
Web output	954
Optimizing a template	954
Scripts	954

Images	955
Generating Print output	956
Saving Printing options in Print Presets	957
Connect Printing options that cannot be changed from within the Printer Wizard	957
Print Using Standard Print Output Settings	958
Print Using Advanced Printer Wizard	959
Adding print output models to the Print Wizard	960
Splitting printing into more than one file	961
Print output variables	961
Generating Fax output	970
Generating Tags for Image Output	971
Generating Email output	973
Email output settings in the Email context and sections	974
Generating Email output from Connect Designer	974
Generating Email output from Workflow	976
Using an ESP with PlanetPress Connect	976
Generating Web output	981
Attaching Web output to an Email template	982
Generating Web output from Workflow	983
Web output settings in the Web context and sections	983
Overview	984
Connect 1.8 General Enhancements and Fixes	987
Connect 1.8 Performance Related Enhancements and Fixes	991
Connect 1.8 Designer Enhancements and Fixes	992
Connect 1.8 DataMapping Enhancements and Fixes	997
Connect 1.8 Output Enhancements and Fixes	1000
Capture OnTheGo (COTG) Enhancements and Fixes	1005
Workflow 8.8 Enhancements and Fixes	1006
Known Issues	1011
Previous Releases	1015
Overview	1015
Connect 1.7.1 General Enhancements and Fixes	1018
Connect 1.7.1 Designer Enhancements and Fixes	1023
Connect 1.7.1 DataMapping Enhancements and Fixes	1031
Connect 1.7.1 Output Enhancements and Fixes	1034
Workflow 8.7 Enhancements and Fixes	1042
Known Issues	1045

Overview	1049
OL Connect Send	1052
Connect 1.6.1 General Enhancements and Fixes	1054
Connect 1.6.1 Designer Enhancements and Fixes	1055
Connect 1.6.1 DataMapping Enhancements and Fixes	1056
Connect 1.6.1 Output Enhancements and Fixes	1056
Connect Workflow 8.6 Enhancements and Fixes	1058
Known Issues	1060
Overview	1065
Connect 1.5 Designer Enhancements and Fixes	1066
Connect 1.5 DataMapping Enhancements and Fixes	1070
Connect 1.5 Output Enhancements and Fixes	1070
Connect 1.5 General Enhancements and Fixes	1072
Connect 8.5 Workflow Enhancements and Fixes	1073
Known Issues	1074
Overview	1078
Connect 1.4.2 Enhancements and Fixes	1080
Connect 1.4.1 New Features and Enhancements	1080
Connect 1.4.1 Designer Enhancements and Fixes	1082
Connect 1.4.1 DataMapping Enhancements and Fixes	1084
Connect 1.4.1 Output Enhancements and Fixes	1084
Connect 8.4.1 Workflow Enhancements and Fixes	1085
Known Issues	1085
Legal Notices and Acknowledgements	1090
Copyright Information	1096

Welcome to PlanetPress Connect 1.8

Note

Since we are always looking for new ways to make your life easier, we welcome your questions and comments about our products and documentation. Use the feedback tool at the bottom of the page or shoot us an email at doc@ca.objectiflune.com.

PlanetPress Connect is a series of tools designed to optimize and automate customer communications management. They work together to improve the creation, distribution, interaction and maintenance of your communications.

The PlanetPress Connect **Datamapper** and **Designer** are designed to create output for print, email and the web within a single template and from any data type, including formatted print streams. Output presets applied outside the design phase make templates printing device independent.

The **Designer** has an easy-to-use interface that makes it possible for almost anyone to create multi-channel output. More advanced users may use native HTML, CSS and JavaScript.

PlanetPress Connect also includes a process automation server, called **Workflow**. It is capable of servicing response form web pages and email to provide interactive business communications.

PlanetPress Connect can create documents for tablets and mobile devices that run a free **CaptureOnTheGo** App. Users with a CaptureOnTheGo subscription can then download documents to their own devices, interact with them and send the captured data back to PlanetPress for conversion into additional documents or workflows.

This online documentation covers **PlanetPress Connect** version 1.8.

Setup And Configuration

This chapter describes the PlanetPress Connect installation and the different considerations that are important in regards to the installation and use of PlanetPress Connect.

- "System and Hardware Considerations" below
- "Installation and Activation" on page 28
- "Known Issues" on page 79
- "Server Configuration Settings" on page 84
- [Uninstalling](#)

System and Hardware Considerations

There are a variety of considerations to be aware of. These are documented in the following pages:

- "Antivirus Exclusions" below
- "Database Considerations" on page 17
- "Environment Considerations" on page 22
- "Language and Encoding Considerations" on page 24
- "Network Considerations" on page 24
- "Performance Considerations" on page 25
- "System Requirements" on page 27

Antivirus Exclusions

The information on this page is designed to assist IT managers and IT professionals decide what anti-virus strategy to follow with consideration to PlanetPress Connect and their internal requirements and needs. This page describes the mode of operation and the files and folders used by PlanetPress Connect as well as the files, folders and executables that are recommended to be ignored for best possible performance and to avoid issues caused by antivirus file locks.

IT managers and IT professionals then may decide the anti-virus strategy to follow for their internal requirements and needs depending on the statements outlined herein.

Directories and folders

Main installation folder

All Connect applications are installed under an arbitrarily selectable main folder. We will speak of the "Installation Target" in the following. This installation target will hold the executables and required files and folders for the operation of the whole product suite. All these files and folders are static after their installation. It depends on the company virus protection strategy, if such files and folders will be monitored or not. A virus protection on these files and folders should, however, not have a big – if even any – impact on the performance of the Connect suite.

Working folders

Working folders for Connect are created and used on a per-user-basis under the respective user's profile folder, accessible on Windows with the standardized system variable `%USERPROFILE%` in the subfolder "Connect". Working folders are:

- **`%USERPROFILE%\Connectfilestore`**: This folder will hold non-intermediate files for the operation of Connect. Files in this folder will be used frequently, but not with a high frequency. Supervising this folder with a virus protection system should not have too much of an impact on the speed of the whole Connect suite.
- **`%USERPROFILE%\Connectlogs`**: As the name implies, log files are created and updated here. These log files are plain text files. Virus protection may have an impact on the speed of the whole Connect suite.
- **`%USERPROFILE%\Connecttemp`**: Storage folder for temporary data, usually intermittent files in multiple folders. Virus protection on this folder and its subfolders may have a serious impact on the performance of Connect.
- **`%USERPROFILE%\Connectworkspace`**: Usually containing settings and helper files and folders. Supervising this folder with a virus protection system should not have too much of an impact on the speed of the whole Connect suite.

Database 1

Depending on the components installed, a database instance is created in the system temp folder of Windows. This folder is accessible via the standardized system variable `%TMP%`. Usually, folders holding such temporary files and folders should be excluded from a virus protection, because this influences the overall performance of the whole system at all. However

the responsible person for the computer protection has to decide about the monitoring of such temporary folders following the company guidelines.

Database 2

Another database instance for Connect will be hold and used under the folder, which is intended to hold data, accessible by and for all users. The path to this folder is stored in the standardized system variable %PROGRAMDATA%. The Connect database instance is located in the subfolder "Connect\MySQL".

As this database will be in extremely strong usage, virus protection on this folder and its subfolders may have a serious impact on the performance of Connect.

Database Considerations

This page describes the different considerations and pre-requisites for the database back-end used by PlanetPress Connect, whether using the MySQL instance provided by the installer, or pre-existing (*external*) instance.

Using the MySQL Instance from the Installer

The MySQL Instance provided in the [Installation Wizard](#) is already pre-configured with options to provide the most stable back-end setup.

These are the specific options that have been changed in our version of "*my.ini*":

- **max_connections = 200** : PlanetPress Connect uses *a lot* of database connections. This number ensures that even in high volume environments, enough connections will be available.
- **max_allowed_packet = 500M** : In some implementations, especially when using Capture OnTheGo, large packet sizes are required to allow transferring binary files. This substantial packet size maximum setting ensures that the data received by PlanetPress Connect will be able to be stored within the database.
- **character-set-server = utf8 , collation-server = utf8_unicode_ci , default-character-set=utf8** : These indicate database support for UTF-8/Unicode.

Installing / Updating Connect Using an existing local MySQL instance

If MySQL Server is already present and you wish to use it, the following should be taken into consideration:

- The MySQL account must have access to all permissions using the GRANT Command, including creating databases.
- The database configuration must include the options detailed in the "Using the MySQL Instance from the Installer" on the previous page topic above.
- The SQL instance must be open to access from other computers. This means the bind-address option should not be set to 127.0.0.1 or localhost.

Warning

If you chose **not** to install the supplied MySQL database, and instead opt for using a pre-existing (*External*) database then you yourself must ensure that the *External* database is accessible to Connect.

Objectif Lune Inc. will take no responsibility for database connections to any but the supplied MySQL database.

Options available within the installer:

- The Configuration page for the local MySQL is displayed.
- MySQL settings are pre-filled with default values if no existing MySQL db configuration is found.
- MySQL settings are pre-filled with existing db configuration settings, if they point to a MySQL db type.

Installing Connect using an existing Microsoft SQL Server instance

If Microsoft SQL Server is already present and you wish to use it, the following should be taken into consideration:

Warning

If you chose **not** to install the supplied MySQL database, and instead opt for using a pre-existing (*External*) database then you yourself must ensure that the *External* database is accessible to Connect.

Objectif Lune Inc. will take no responsibility for database connections to any but the supplied MySQL database.

Note

Since PlanetPress Connect version 1.6 the minimum required version of the MS SQL Server is **SQL Server 2012**.

- When MS SQL is selected, the default values for root user are **sa** and **1433** for the port.
- If db settings from a previous installation are found, the pre-existing settings will be displayed for the matching db type (for MS SQL settings, this will only work if they were created with Server Config Tool 1.5.0 or later, or the Connect installer 1.6.0 or later). If the db type is changed in the configuration page, the default values for this db type will be displayed. If the pre-existing db settings are set to Hsqldb, the default db type selection will be MySQL.
- Selected db settings are stored in the preferences as usual (C:\ProgramData\Objectif Lune\OI Connect\settings\ConnectHostScope\com.objectiflune.repository.eclipselink.generic.preferences)

Updating With No Local MySQL Product

- When updating a Connect installation from 1.5.0 which contains a Server Product but no local MySQL Product, the DB Configuration Page will detect which db type was set before (especially if the db configuration was switched from MySQL to MS SQL using the Server Configuration Tool), and default to those settings.
- On Update from 1.4.2 or earlier, the DB Configuration Page will always default to MySQL connection settings, and if the installation was manually tweaked to connect to MS SQL Server, the user has to switch to "Microsoft SQL Server" type and enter connection details again.

When modifying Connect

- If local MySQL is removed from an installation, the DB Configuration page will offer additionally the **Microsoft SQL Server** db type with respective default values.

- If local MySQL is added to an installation, the usual MySQL Configuration page with default values will be displayed.

If the user has installed the Installer Supplied MySQL and then switches to an *external* Microsoft SQL by using the Server Configuration Tool, the supplied MySQL cannot be switched off. By design the installer adds a service dependency between Connect Server and the supplied MySQL service.

Note

The Microsoft SQL selection capability will be available only with 1.6 version and upwards.

To remove this dependency the user needs to do the following

1. Have a foreign Microsoft SQL running, ready for use with Connect Server.
2. Use the **Server Configuration Tool** "Database Connection preferences" on page 700 to switch the database to Microsoft SQL.
3. Re-start the Connect Server Service, so that the modifications become active.
4. Counter check that everything is working properly with Microsoft SQL.
5. Open a command-line prompt with full administration rights.
6. Enter the command `sc config OLCConnect_Server depend= /`. This removes the dependency.
Please be aware: The key word `depend` must be followed immediately by the equal sign, but between the equal sign and the forward slash there must be a space.
Additional information can be found here: <http://serverfault.com/questions/24821/how-to-add-dependency-on-a-windows-service-after-the-service-is-installed#228326>.
7. After the dependency has been removed, it is possible to stop the supplied MySQL service (OLConnect_MySQL).

Warning

If a Connect 1.5 user wants to use Microsoft SQL instead of MySQL for the Connect Server, there are several points to be taken care of:

- **IF** there should possibly be available some foreign MySQL instance, which could be used intermediately, then this should be selected during the setup. This ensures, that no stuff gets installed. Otherwise the supplied MySQL needs to be installed and the switch to Microsoft SQL needs to be done as outlined above.
- It is not possible to uninstall the supplied MySQL in this case via a Connect 1.5 modify.

Important

If a Server Product and a MySQL Product were selected to be installed on Connect 1.5.0, and then the Server Configuration Tool is used to switch the database used by the Server to an external Microsoft SQL, then the Update to 1.6 requires an extra step. The procedure is as follows:

1. Run the **Update to Connect 1.6**. This will assume the local MySQL database needs to be updated and configured, so the user has to enter a root password on the MySQL Configuration Page (can be any password matching Connect security rules).
2. After the update, the **Connect 1.6 Setup** needs to be run once more to modify Connect.
3. On the **Product Selection** page, now the MySQL product can be unselected.
4. When stepping forward in the Wizard, the DB Configuration page will be displayed which allows to configure the Microsoft SQL Server with appropriate settings.

After this modification, the local MySQL is removed, and also the service dependency from Server to MySQL is removed.

Note

If Connect was initially installed not containing the local MySQL product (i.e. on 1.5 installation an external MySQL was configured as database), then the Update to 1.6 will allow to select either external MySQL or external Microsoft SQL on the DB Configuration Page.

Environment Considerations

Virtual Machine Support

PlanetPress Connect supports VMWare Workstation, VMWare Server, VMWare Player, VMWare ESX (including VMotion), Microsoft Hyper-V and Microsoft Hyper-V/Azure infrastructure environments as software installed on the Guest operating system.

Warning

Copying (duplicating) a Virtual Machine with Connect installed and using both images simultaneously constitutes an infringement of our End-User License Agreement.

Note

While some virtual machine environments (from VMWare and Microsoft) are supported, other virtual environments (such as Parallels, Xen and others) are not supported at this time.

Terminal Server/Service

PlanetPress Connect does not support Terminal Server (or Terminal Service) environment as possible under Windows 2000, 2003 and 2008. This is to say, if Terminal Service is installed on the server where PlanetPress Connect is located, unexpected behaviours may occur and will not be supported by Objectif Lune Inc.. Furthermore, using PlanetPress Connect in a Terminal Service environment is an infringement of our End-User License Agreement.

Remote Desktop

Tests have demonstrated that PlanetPress Connect can be used through Remote Desktop. It is however possible that certain combination of OS could cause issues. If problems are encountered, please contact OL Support and we will investigate.

PlanetPress Connect 1.3 and later have been certified under Remote Desktop.

64-bit Operating Systems

PlanetPress Connect is a 64-bit software and can only be installed on 64-bit operating systems.

Antivirus Considerations

- Antivirus software may slow down processing or cause issues if they are scanning in temporary folders or those used by PlanetPress Connect. Please see KB-002: Antivirus Exclusions for more information.
- Antivirus software might interfere with installation scripts, notably a vbs script to install fonts. McAfee, in particular, should be disabled temporarily during installation in order for MICR fonts to install and the installation to complete successfully.

Windows Search Indexing Service

Tests have concluded that the Windows Search service, used to provide indexing for Windows Search, can interfere with Connect when installing on a virtual machine. If the installation hangs during the last steps, it is necessary to completely disable this service during installation.

- Click on Start, Run.
- Type in **services.msc** and click OK.
- Locate the **Windows Search** service and double-click on it.
- Change the **Startup Type** to **Disable**, and click **Stop** to stop the service.
- Try the installation again.
- Once completely, you may re-enable the service and start it.

Commandline switches and .ini entries

PlanetPress Connect is intended to work stably and reliably, based on Java and the Eclipse framework. To ensure this reliability and robustness, many Java and Eclipse parameters have been tested and tuned, which is reflected in the respective .ini entries and the used command line switches. A collection of valuable settings has been elaborated and found its entry in PlanetPress Connect “good switches list” (called the “whitelist”).

The protection of the end user’s system is one of our main goals and therefore we have implemented a very strict verification mechanism, which ensures, that only these whitelisted ini entries and command-line switches are accepted, when one of Connect components is started and run. Please be therefore advised, that any non-whitelisted ini entry or command-line switch will be accepted and will - if tried to be used - lead to the respective application’s “sudden death”. If you should encounter such a behaviour then please double-check your Connect log file/s for respective entries.

Language and Encoding Considerations

Please note the following considerations:

- **Language:**

- PlanetPress Connect is currently offered in several languages. These languages can be switch between via the Preferences dialog. The current languages include:
 - English
 - French
 - German
 - Spanish
 - Italian
 - Korean
 - Portuguese
 - Chinese (Simplified)
 - Chinese (Traditional)
 - Japanese.

The default language is English.

The PlanetPress Connect help system (this document) is currently only available in English.

- **Encoding:**

- Issues can sometimes be encountered in menus and templates when running PlanetPress Connect on a non-English operating system. These are due to encoding issues and will be addressed in a later release.

Network Considerations

The following should be taken into consideration in regards to network settings and communications

- If a local proxy is configured (in the **Internet Explorer Options** dialog, the option **Bypass proxy server for local addresses** must be checked, or some features depending on local communication will not work.

Firewall/Port considerations

For Firewall/Port considerations, please see this article in the Knowledge Base: [Connect Firewall/Port Configuration](#)

Performance Considerations

This page is a comprehensive guide to getting the most performance out of PlanetPress Connect as well as a rough guideline to indicate when it's best to upgrade.

Performance Analysis Details

In order to get the most out of PlanetPress Connect, it is important to determine how best to maximize performance. The following guidelines will be helpful in extracting the best performance from PlanetPress Connect before looking into hardware upgrades or extra PlanetPress Connect performance packs.

- **Job Sizes and Speed:** In terms of pure output speed, it's important to first determine what job size is expected, and adjust "[Scheduling Preferences](#)" on page 85 accordingly. The basic rules are:
 - If processing a small number of very large records (when each individual record is composed of a large number of pages), more instances with an equal amount of speed units is better. For hardware, RAM and Hard Drive speeds are most important, since the smallest divisible part (the record) cannot be split on multiple machines or even cores.
 - If creating a very large number of small records (hundreds of thousands of 2-3 page individual records, for instance), a smaller number of instances with a large number of speed units would be better. As for hardware, then the number of cores becomes critical, whereas RAM and hard drive are secondary. Performance Packs, as well as the MySQL instance being separate, would be helpful if your most powerful machine starts struggling.
 - Mix and match. For example, one instance prioritized for large jobs and the rest for smaller, quicker jobs. Or the contrary. Or, whatever you want, really.
- **RAM Configuration:** By default, each instance of the Merge Engine and Weaver Engine is set to use 640MB of RAM. This means that regardless of speed units, if not enough memory is available, output speed might not be as expected. Assuming that the machine itself is not running any other software, the rule of thumb is the following: The total number of used memory in the machine should be pretty much the maximum available (around 95%).

For each engine, it's necessary to modify the .ini file that controls its JAVA arguments. Edit as follows:

- For the Merge Engine: see C:\Program Files\Objectif Lune\OL Connect\MergeEngine\Mergeengine.ini
- For the Weaver Engine: see C:\Program Files\Objectif Lune\OL Connect\weaverengine\Weaverengine.ini
- The parameters are -Xms640m for the minimum RAM size, -Xmx640m for the maximum RAM size. Explaining Java arguments is beyond the scope of this document. Please read references [here](#), [here](#) and [here](#) for more details (fair warning: these can get pretty technical!).
- **Template and data mapping optimization:** Some functionality offered by the DataMapper and Designer modules is very useful, but can cause the generation of records and of contents items to slow down due to their nature. Here are some of them:
 - **Preprocessor and Postprocessor scripts:** Manipulating data using a script may cause delays before and after the data mapping action has actually taken place, especially file conversion and data enrichment from other sources.
 - **Loading external and network resources:** In Designer, using images, javascript or css resources located on a slow network or on a slow internet connection will obviously lead to a loss of speed. While we do our best for caching, a document with 100,000 records which queries a page that takes 1 second to return a different image each time will, naturally, slow output generation down by up to 27 hours.
 - **External JavaScript libraries:** While loading a single JavaScript library from the web is generally very fast (and only done once for the record set), actually running a script on each generated page can take some time. Because yes, JavaScript will run for each record, and often take the same time for each record.
 - **Inefficient selectors:** Using very precise ID selectors in script wizards can be much faster than using a text selector, especially on very large documents. See also: "Use an ID as selector" on page 636.
 - **Complex scripts:** Custom scripts with large, complex or non-optimized loops can lead to slowing down content creation. While it is sometimes difficult to troubleshoot, there are many resources online to help learn about JavaScript performance and coding mistakes. [Here](#), [here](#), and [here](#) are a few. Note that most resources on the web are about JavaScript in the *browser*, but the greatest majority of the tips do, indeed, apply to scripts in general, wherever they are used.

High-performance hardware

The following is suggested when processing speed is important. Before looking into Performance Packs to enhance performance, ensure that the below requirements are met.

- **MySQL Database on a separate machine.** MySQL's main possible bottleneck is file I/O, and as such a high-performance setup will require this server to be on a separate machine, ideally with a high-performance, low-latency hard drive. A Solid State Drive (SSD) would be recommended.
- **High-Quality 16+ GB Ram.** This is especially true when working with many server instances ("speed units") running in parallel. The more parallel processing, the more RAM is recommended.
- **4 or 8 physical cores.** We're not talking Hyper-Threading here, but physical cores. Hyper-Threading is great with small applications, but the overhead of "switching" between the virtual cores, and the fact that, well, they're virtual, means the performance is much lesser on high-power applications such as OL Connect. In short, a dual-core processor with Hyper-Threading enabled is not equivalent to a quad-core processor.
- **Preferably use a physical, non-virtualized server.** VMWare servers are great for reducing the numbers of physical machines in your IT space, but they must share the hardware between each other. While you can create a virtual machine that seems as powerful as a physical, it will still be sharing hardware with any other virtual machines, and this will adversely affect performance.

System Requirements

These are the system requirements for PlanetPress Connect 1.8

Operating System (64-bit only)

- Microsoft Windows 2008/2008 R2 Server
- Microsoft Windows 2012/2012 R2 Server
- Microsoft Windows Vista
- Microsoft Windows 7
- Microsoft Windows 8.1
- Microsoft Windows 10 (Pro and Enterprise versions only)

Note

Windows 8.0, Windows XP, Windows 2003 and older versions of Windows are not supported by PlanetPress Connect.

Minimum Hardware Requirements

- NTFS Filesystem (FAT32 is not supported)
- CPU Intel Core i7-4770 Haswell (4 Core)
- 8GB RAM (16GB Recommended)
- Disk Space: At least 10GB (20GB recommended)

Note

For tips and tricks on performance, see "Performance Considerations" on page 25.

Installation and Activation

This topic provides detailed information about the installation and activation of PlanetPress Connect 1.8.

Note

A PDF version of this guide is available for use in offline installations. [Click here to download it.](#)

PlanetPress Connect 1.8 is comprised of 2 different installers: one for the PlanetPress Connect software and one for PlanetPress Workflow 8.

Where to obtain the installers

The installers for PlanetPress Connect 1.8 and PlanetPress Workflow 8 can be obtained on DVD or downloaded as follows:

- If you are a **Customer**, the installers can be downloaded from the Objectif Lune Web Activations page: <http://www.objectiflune.com/activations>
- If you are a **Reseller**, the installers can be downloaded from the Objectif Lune Partner Portal: <http://extranet.objectiflune.com/>

Installation - important information

For important information about the Installation, including requirements and best practices, please see the following topics:

- [Installation Prerequisites](#)
- [User accounts and security](#)
- [The importance of User Credentials when installing and running Connect](#)
- [Migrating to a new computer](#)

Installation - "How to" guides

For information on how to conduct the installation itself, chose from the following topics:

- [Installation](#)
- [Silent Installation](#)
- [Installation on machines without Internet access](#)

Activation

For information on licensing, please see [Activating your license](#).

Installation Prerequisites

- Make sure your system meets the [System requirements](#).
- PlanetPress Version 1.8 can be installed [under a regular user account with Administrator privileges](#).
- PlanetPress **must** be installed on an NTFS file system.
- PlanetPress requires **Microsoft .NET Framework 3.5** already be installed on the target system.

- In order to use the **automation features** in Version 1.8, **PlanetPress Workflow 8** will need to be installed.
This can be installed on the same machine as an existing PlanetPress® Suite 7.6 installation or on a new computer.
For more information, please see [Information about PlanetPress Workflow 8](#).
- As with any JAVA application, the more RAM available, the faster PlanetPress will execute!

Users of Connect 1.1

In order for users of PlanetPress Connect 1.1 to upgrade to any later version through the Update Manager it is necessary to install a later version (1.1.8 or later) of the Objectif Lune Update Client.

If you do not have such a version installed already, the next time you run your Update Client it will show that there is an update available of itself to Version 1.1.8 (or later).

Simply click on the download button in the dialog to install the new version of the Update Client. Note that it is no problem to run the update while the Client is open. It will automatically update itself.

Once you have done this, PlanetPress Connect 1.8 will become available for download.

Note

From PlanetPress Connect Version 1.2 onwards, the new version (1.1.8) of the Update Client is included by default with all setups.

Users of Connect 1.0

Users of this Connect version 1.0 cannot upgrade directly to Version 1.8. This is because Connect Version 1.0 is a 32 bit version of Connect.

Users must first upgrade to Version 1.1 and from there upgrade to Version 1.8

If you are updating manually you must first upgrade to Version 1.1 before installing 1.8. If you attempt go directly from Version 1.0 to Version 1.8 the installation will fail.

Also see "Users of Connect 1.1" above for extra information about updating from that version.

User accounts and security

Permissions for PlanetPress Connect Designer

PlanetPress Connect Designer does not require any special permissions to run besides a regular program. It does not require administrative rights and only needs permission to read/write in any folder where Templates or Data Mapping Configurations are located.

If generating Print output, PlanetPress Connect Designer requires permission on the printer or printer queue to send files.

Permissions for PlanetPress Connect Server

The PlanetPress Connect Server module, used by the *Automation* module, requires some special permissions to run. These permissions are set during installation, in the *Engine Configuration* portion of the [Installation Wizard](#), but it can also be configured later by modifying permissions for the service. To do this:

- In Windows, open the Control Panel, Administrative Tools, then Services (this may depend on your operating system).
- Locate the service called `Serverengine_UUID`, where UUID is a series of characters that depend on the machine where the software is installed.
- Right-click on the service and select *Properties*.
- In the *Connection* tab, define the account name and password that the service should use. This can be a local account on the computer or an account on a Windows Domain. The account must have administrative access on the machine. It should also correspond to the user account set up in *PlanetPress Workflow*.

The Importance of User Credentials on Installing and Running PlanetPress Connect

OL Connect and required credentials depends heavily on the Connect component and respective tasks and what sort of user credentials are needed.

First of all, it is important to distinguish between installation and run-time

Installation

The Connect installer puts all required files, folders, registry entries and much more to their correct places and locations. As many of these locations are protected against malicious accesses, that very user under whose context the Connect installation is started and running, needs very extensive rights on the respective computer. This user must belong to the Local Administrators group on that machine. Here are some required capabilities, this user:

- Must be able to write into the "Programs" folder.
- Must be allowed to check for existing certificates and must also be allowed to install new ones into the global certificate store on that machine.
- Must be able to write into HKLM and any subtree of it in the registry.
- Must be able to INSTALL, START and RUN services and also to MODIFY service settings.
- Must be known in the network the machine belongs to and must also need to be able to use shared network resources like shared drives and/or printers etc.

This list may not be complete, but it gives the extent of the requirements. Generally, the local administrator of the machine will have all these credentials, but there may exist network restrictions and policies, which will block one or more of these capabilities. In such cases, the respective network administrator should provide a valid user account for the installation.

User Account

The user account shall be used to later RUN one of the Connect Server flavors (Server or Server Extension). This dedicated user account has to be entered on the respective installer dialog page and must be allowed to START, STOP and RUN services on this machine. This is different from the credentials of the installation user account, which additionally requires the right to INSTALL services. Please be aware of this fact!

Additionally, the Server user must be able to access any network resources that are required for OL Connect to function properly. This includes e.g. additional drives, printers, scanners, other computers and, where appropriate, internet resources, URLs, mail servers, FTP servers, database servers and everything else planned to be used for the intended operation of Connect. The Server user is the run-time user.

Connect Components

Usually, a standard end user will only be facing Connect Designer and maybe the License Activation Tool. Designer this does not require administrator rights. Either everything required

to create documents or also to run some tasks will be already available (installed by the installer) or be accessible in a way, where no specific credentials are required. However some tasks like starting an email campaign will possibly require a respective account at a mail server. But this has generally nothing to do with the credentials of the Designer user.

Activation Tool

To run the Software Activation Tool, administrator rights are required because this tool needs to write the license file in one of the protected folders of Windows. The tool will however allow to restart it with respective credentials if required.

MySQL

MySQL database service is installed by the install user (thus again the requirement of installing, starting, running and modifying services). Once running it will just work.

Merge and Weaver Engines

These components do run under the Designer (if only Designer is installed) or the Server / Extension service(s) and inherit the rights of their parent application.

Server (Extension) Configuration Tool

This component needs to access the settings of the Server. As these are stored and read by the Server, it should be clear that the user used to run the Configuration tool should be the same as the Server Service user as explained above.

Installing PlanetPress Connect on Machines without Internet Access

Installing PlanetPress Connect 1.8 in offline mode requires some extra steps. These are listed below.

GoDaddy Root Certificate Authority needs to be installed.

In order to install PlanetPress Connect it is necessary for the GoDaddy Root Certificate Authority to be installed (G2 Certificate) on the host machine and for this to be verified online. When a machine hosting the installation does not have access to the Internet, the installation will fail because the verification cannot be performed. To solve this problem one must first ensure that all Windows updates have been installed on the host machine. Once the Windows

updates are confirmed as being up to date, then complete the following steps:

1. Go to <https://certs.godaddy.com/repository> and download the following two certificates to copy to the offline machine:
 - GoDaddy Class 2 Certification Authority Root Certificate - G2 - the file is gdroot-g2.crt
 - GoDaddy Secure Server Certificate (Intermediate Certificate) - G2 - the file is gdig2.crt
2. Install the certificates: Right mouse click -> Install Certificate, and follow the steps through the subsequent wizard.
3. Now copy the PlanetPress Connect installer to the offline machine and start the installation as normal

Windows certificate validation - Certificate Revocation List retrieval should be switched off

For your security Objectif Lune digitally signs all relevant files with our own name and certificate. The integrity of these files is checked at various times by different, context related, methods. One of these checks, done during the installation process, uses the Windows certificate validation check. .

The Windows certificate validation process not only checks the integrity of a file against its signature, but also usually checks if the certificate itself is still valid. That check is done against the current Certificate Revocation List (CRL), which needs to be retrieved from the internet. However, if the machine in question does not have internet access, the retrieval of the CRL must fail, which will lead to subsequent validation issues.

To circumvent such issues it is **highly recommended** to switch off the CRL retrieval prior to installing Connect on machines without internet access. There is no security risk associated with this, as the CRLs would never be retrievable without internet access, anyway. Advantage of the switch will not only be found during the installation and operation of Connect, but also in some speed improvements for any application which use signed binaries.

To switch off CRL retrieval on the computer, complete the following steps:

1. Open the “Internet Options” via the Control Panel
2. Select the “Advanced” tab and scroll down to “Security” node.
3. Uncheck the entry “Check for publisher’s certificate revocation” under that node.

4. Click the OK button to close the dialog.
5. Re-start the computer.

Installation Wizard

Starting the PlanetPress Connect installer

The PlanetPress Connect installer may be supplied as an ISO image or on a DVD.

- If an ISO image, either burn the ISO onto a DVD or unzip the contents to a folder (keeping the folder structure)
- If on a DVD, either insert the DVD and initiate the installation from there or copy the contents to a folder (keeping the folder structure)

Navigate to the PlanetPress_Connect_Setup_x64.exe or and double-click on it. After a short while the Setup Wizard will appear as a guide through the installation steps.

Note

PlanetPress Connect **requires** prior installation of Microsoft .NET Framework 3.5.

Please refer to <https://www.microsoft.com/en-us/download/details.aspx?id=21> for more details on how to install Microsoft .NET Framework 3.5, if this is not already done.

Note

If the same version of PlanetPress Connect is already installed on the target machine, you will be presented with options to either *Uninstall* or *Modify* the existing instance.

If **Modify** is selected, the standard installation Wizard sequence will be followed, but with all options from the existing installation selected.

Selecting the required components

After clicking the Next button, the component selection page appears, where the different components of PlanetPress Connect can be selected for installation. Currently, the following are available:

- **PlanetPress Connect Designer:** The Designer module (see "The Designer" on page 302). It may be used as a standalone with no other installed modules, but it will not have certain capabilities such as automation and commingling.
- **PlanetPress Connect Server:** The Server back-end giving capabilities such as automation, commingling, picking. It saves all entities generated from the Automation module into a database for future use.
- **MySQL Product:** Install the supplied MySQL database used by PlanetPress Connect. The database is used for referencing shared and temporary Connect files, as well as for sorting temporarily extracted data, and the like.
A pre-existing MySQL or Microsoft SQL server (referred to as an **external** database, in this documentation) *could* be used for the same purpose, however.
The *external* database could reside on either the same computer or on a separate server. If you wish to make use of an *external* database, please make sure the **MySQL Product** option is not selected.

Warning

If you chose not to install the supplied MySQL database, and instead opt for using a pre-existing *external* database then you must ensure that your *external* database is accessible to Connect, yourself. Objectif Lune Inc. will take no responsibility for database connections to any but the supplied MySQL database.
See "Database Considerations" on page 17 for more information about setting up *external* databases.

- **Installation Path:** This is the location where modules are to be installed.

The installer can also calculate how much disk space is required for installing the selected components as well as how much space is available:

- **Disk space required:** Displays the amount of space required on the disk by the selected components.
- **Disk space available on drive:** Displays the amount of space available for installation on the drive currently in the Installation Path.
- **Recalculate disk space:** Click to re-check available disk space. This is useful if space has been made available for the installation while the installer was open.

- **Source repository location:** Displays the path where the installation files are located. This can be a local drive, installation media, or a network path.

Selection Confirmation

The next page confirms the installation selections made. Click **Next** to start the installation itself.

End User License Agreement

The next page displays the [End User License Agreement](#), which needs to be read and accepted before clicking **Next**.

Configuring Supplied Database Connection

The **Default Database Configuration** page appears if the supplied *MySQL Product* module was selected for installation in the *Product Selection* screen. It defines the administrative password for the MySQL server as well as which port it uses for communication.

The installer will automatically configure the *Connect Server* to use the supplied password and port.

- **MySQL user 'root' Password:** Enter the password for the 'root', or administration account, for the MySQL server. The password must be at least 8 characters long and contain at least one of each of the following:
 - a lower case character (a, b, c ...)
 - an upper case character (A, B, C ...)
 - a numeric digit (1, 2, 3 ...)
 - a punctuation character (@, \$, ~ ...)

For example: "This1s@K"

Note

When updating from an earlier Connect version, the appropriate MySQL password **must** be entered or the update will fail.

If the password is subsequently forgotten, then the MySQL product must be uninstalled and its database deleted from disk before attempting to reinstall.

- **Confirm 'root' Password:** Re-enter to confirm the password. Both passwords must match for installation to continue.
- **TCP/IP Port Number:** The port on which MySQL will expect, and respond to, requests. A check is run to confirm whether the specified TCP/IP Port Number is available on the local machine. If it is already being used by another service (generally, an existing MySQL installation), the number is highlighted in red and a warning message is displayed at the top of the dialog.

Note

The MySQL Product controlled by the *OLConnect_MySQL* service communicates through port 3306 by default.

- **Allow MySQL Server to accept non-local TCP connections:** Click to enable external access to the MySQL server.

Note

This option is required if MySQL Server will need to be accessed from any other machine.

It is also required if the MySQL database is on a separate machine to PlanetPress Connect.

Tip

This option may represent a security risk if the machine is open to the internet.

It is heavily recommended that your firewall is set to block access to port 3306 from external requests.

Configuring External Database Connection

The **Database Connection** page appears if the supplied MySQL Product module was not selected for installation. This page is for setting up the connection to the existing External database.

- **Database Configuration:** Select the database type to use for the PlanetPress Connect Engine. Currently only MySQL and Microsoft SQL Server are supported.
- **Administrator Username:** Enter the username for a user with administrative rights on the database. Administrative rights are required since tables need to be created in the database.
If accessing a database on a different machine, the server must also be able to accept non-local TCP connections, and the username must also be configured to accept remote connection. For example, the "root" MySQL user entered as root@localhost is not allowed to connect from any other machine than the one where MySQL is installed.
- **Administrator Password:** Enter the password for the above user. The appropriate MySQL password **must** be entered or the Connect installation will fail.
- **TCP/IP Port Number:** Enter the port on which the database server expects connections. For MySQL, this is **3306** by default.
For MS SQL it is **1433** by default.
- **Database Host Name:** Enter the existing database server's IP or host name.
- **Server Schema/Table:** Enter the name of the MySQL database into which the tables will be created. The Connect standard name is "objectiflune".
- **Test Connection** button: Click to verify that the information provide into previous fields is valid by connecting to the database.

Note

This test does not check whether the remote user has READ and WRITE permissions to the tables under the objectiflune schema. It is solely a test of database connectivity.

PlanetPress Connect Server Configuration

The **Server Configuration** page is where the *Connect Server* component is configured.

The **Connect Server** (Master) settings are as follows:.

- **Run Server as:** Defines the machine username and password that the **PlanetPress Connect Server** module's service uses.

Note

The "Server Security Settings" on page 90 dialog can only ever be executed from the user specified here.

- **Username:** The account that the service uses to login. If the machine is on a domain, use the format domain\username.
This account must be an existing Windows profile with local administrator rights.
- **Password:** The password associated with the selected user.
- **Validate user** button: Click to verify that the entered username and password combination is correct and that the service is able to login.
This button *must* be clicked and the user validated before the **Next** button becomes available.

Click **Next** to start the actual installation process. This process can take several minutes.

Completing the installation

This screen describes a summary of the components that have been installed.

- **Configure Update Check** checkbox: This option is enabled by default. It causes the **Product Update Manager** to run after the installation is complete. This allows configuring PlanetPress Connect to regularly check for entitled updates.
Note: this checkbox may not be available in the event that an issue was encountered during the installation.
- **Show Log...** : If an issue was encountered during the installation, click this button to obtain details. This information can then be provided to Objectif Lune for troubleshooting.
- When ready, click the **Finish** button to close the installation wizard, and initialize the Product Update Manager, if it was selected.

The Product Update Manager

If the **Configure Update Check** option has been selected, the following message will be displayed after clicking "Finish" in the setup:

Click “Yes” to install or open the Product Update Manager where the frequency with which the updates can be checked and a proxy server (if required) can be specified.

Note: if the Product Update Manager was already installed by another Objectif Lune application, it will be updated to the latest version and will retain the settings previously specified.

Select the desired options and then click **OK** to query the server and obtain a list of any updates that are available for your software.

- Note that the Product Update Manager can also be called from the “Objectif Lune Update Client” option in the Start menu.
- It can be uninstalled via Control Panel | Programs | Programs and Features.

Product Activation

After installation, it is necessary to activate the software. See [Activating your license](#) for more information.

Note

Before activating the software, please wait 5 minutes for the database to initialize. If the software is activated and the services rebooted too quickly, the database can become corrupted and require a re-installation.

Running connect installer in Silent Mode

PlanetPress Connect can be installed in a so called "silent mode" to allow an automated setup during a company wide roll-out or comparable situations. The trigger for the Connect Installer to run in silent mode is a text file with the fixed name **install.properties**, which is located either in the same folder as the PlanetPress_Connect_Setup_x86_64.exe or in the unpacked folder of the **installer.exe**.

Note

Only the installation can be run silently. **Silent Mode** does not apply to uninstalling, modifying, or updating Connect. Any previous version of Connect must be uninstalled before using the Silent Installer.

The required properties file has the following attributes:

- Comment Lines, starting with # (e.g. # The options to configure an external database)
- Key = Value pairs (e.g. install.product.0 = Connect Designer)

For supported keys, please refer to the next paragraph.

Note

install.properties file notation must follow common configuration rules. Please refer to [Properties files](#) for more details.

Required and optional properties

Required properties depend on the specified product. Only fields related to that specified product must be entered. If no product is mentioned, properties must be specified for all valid Connect products.

Here is an example of an **install.properties** file.

```
# Verbose logging
logging.verbose = true

# Product selection
install.product.0 = Connect Designer
install.product.1 = Connect Server

# Server settings
server.runas.username = Localadmin
server.runas.password = admin

# Database configuration
database.type = mysql
database.host = 192.168.116.10
database.port = 3308
database.username = root
database.password = admin
database.schema = my_ol
```

Verbose logging (optional)

By default, the **Silent Installer** will log the same way as the GUI installer. That means logging of error and warnings, and certain information during database configuration. A more verbose logging can be switched on by using **logging.verbose = true**.

Product selection (optional)

By default, if nothing is entered for the products to be installed (install.product.X), **Silent Installer** will install all products which are visible to the user for the respective brand (except for the Server Extension, because only Server or Server Extension can be installed at the same time).

PlanetPress defaults

```
install.product.0 = Connect Designer  
install.product.1 = Connect Server  
install.product.2 = MySQL Product
```

Note

The values of **install.product** properties must contain the exact product names.

Server configuration (required if Server is selected for install)

For Server, the following properties need to be provided:

```
server.runas.username = <username>  
server.runas.password = <password>
```

Server Extension configuration (required if Server Extension is selected for install)

For Server Extension, the following properties need to be provided:

```
server.runas.username = <username>  
server.runas.password = <password>  
server.master.host = <IP or name>  
server.master.port = <port>  
server.master.authenticate = true or false  
server.master.username = <username for the Connect Server>  
server.master.password = <password>
```

Database configuration

Case 1: MySQL is among the selected Connect products to be installed (new MySQL installation)

If MySQL is selected and there is no previous MySQL configuration on the machine, the following properties should be defined:

```
database.password = <password> (required and must meet the rules)
database.port = <port> (3306 is the default port value)
database.unlocked = true or false (the default value is false, optional)
```

Note

The **unlocked** option should only be used when the database requires an external access.

If the **Silent Installer** runs with the default product selection, MySQL Product is included, and hence the **database.unlocked = true** property may be optionally set if MySQL on this machine is intended to serve as the central database also for remote machines.

If the **Silent Installer** runs with the explicit installation of a stand-alone (install.product.0 = Connect Server), the **database.unlocked** property is irrelevant.

Note

The port will be defined automatically for the MySQL installation. All connect products selected in the Silent Installer will automatically be configured to use the MySQL running under the port defined by the **database.port** property, regardless of the default port 3306 or any other user defined port.

A different port is required if 3306 is already taken on that machine by another application.

Case 2: The Connect Server is selected and the MySQL Product is not selected

In this case, an external database must be configured for the Server (and other Connect products included in the Silent installation) to be used.

2a: Configuring an external MySQL database

To configure an external MySQL database, the following properties should be defined:

```
database.type = mysql (required)
database.host = <host> (default value is localhost, otherwise
required)
database.port = <port> (default value is 3306, otherwise required)
database.username = <username> (default value is root, otherwise
required)
database.password = <password> (required)
database.schema = <schema name> (default value is objectiflune,
optional)
```

2b: Configuring an external Microsoft SQL Server database

Note

Since PlanetPress Connect version 1.6 the minimum required version of the MS SQL Server is **SQL Server 2012**.

To configure an external Microsoft SQL Server database, the following properties should be defined:

```
database.type = Microsoft SQL Server (required)
database.host = <host> (default value is localhost, otherwise
required)
database.port = <port> (default value is 1433, otherwise required)
database.username = <username> (default value is sa, otherwise
required)
database.password = <password> (required)
database.schema = <schema name> (default value is objectiflune,
optional)
```

Repository selection

The Connect installation process requires a repository from which the installer copies (locally) or downloads (online installation) all selected Connect products.

In Silent Installer mode, the installation process looks for the property **product.repository** in the **install.properties** file and then proceeds with the following steps:

1. If the property exists, and its value contains an existing file location with a repository, the installer will attempt to install from that repository.
2. If the property exists, and its value starts with `http://`, the installer will attempt to install from that location. It will fail if no repository can be found at this location.
3. If none of the conditions mentioned in the previous steps are met, the installer will look next for a local "repository" folder (located in the same folder as the running Installer (Setup) executable file). If a repository is found, the installer will attempt to install from that repository.
4. As a last resort, the installer will attempt to install from the default Connect Update Site URL.

Examples

```
product.repository = http://192.168.79.73/Connect/Version_01/repository
product.repository = C:\\iso\\2.0.0.39695_unpacked\\repository
```

Locale definition

It is possible to define the Locale which affects the installation language and installed locale for Connect products by using the following properties in the **install.properties** file:

```
user.language
user.country
```

Locales supported by Connect

The Connect Setup supports a dedicated list of Locales, which is saved in the `preinstall.ini` file. Each entry consists of a language tag and a country tag, formatted by the pattern:

```
<language>-<country>
```

The current list of supported Locales is found below, but it may be enhanced in future releases:

- en-US (English, US)
- de-DE (German, Germany)
- fr-FR (French, France)

- ja-JP (Japanese, Japan)
- zh-CN (Chinese, China)
- zh-HK (Chinese, Hongkong)
- zh-MO (Chinese, Macau)
- zh-TW (Chinese, Taiwan)
- it-IT (Italian, Italy)
- pt-BR (Portuguese, Brazil)
- es-419 (Spanish, Latin America)

Locale selection by defining `user.language` and `user.country`

If both `user.language` and `user.country` are defined in the `install.properties` file, the combination must match exactly one of the supported locales, otherwise the Installer will exit with an error.

For example, `user.language = fr` and `user.country = CA` will cause an error since fr-CA is not in the list of supported Locales.

Locale selection by defining only `user.language`

If only `user.language` is defined in the `install.properties` file, the Installer will attempt to find a Locale in the list which starts with the given language code. The first match is selected for installation. If no match is found, the Installer will exit with an error.

For example:

`user.language = zh`, will result in an installation with the Locale zh-CN

`user.language = no`, will result in an error

Default Locale selection

If neither `user.language` nor `user.country` is defined in the `install.properties` file, the Installer will select a default Locale:

1. If the System Locale is in the list of supported Locales, it will be selected.
2. Otherwise, if there is an entry in the list of supported Locales, which matches the System language, it will be selected (e.g. on a fr-CA system, fr-FR is selected).

3. As last resort, the first Locale in the preinstall.ini is selected (usually that should be en-US).

Getting the exit code of a silent installation

If getting the exit code of a silent installation is desirable, use the following procedure.

1. Create a new local folder on the machine (or VM) on which Connect shall be installed and copy/extract the contents of the Connect ISO into this folder.
2. Open a command prompt with Administrator privileges and cd into this local folder.
3. Run this command to unpack the contents of the Connect Setup executable (as a sample, we use the PReS Connect brand):

```
PReS_Connect_Setup_x86_64.exe -nr -gm2 -InstallPath=".\""
```
4. In the local folder, the repository subfolder should now be located next to the preinstall.exe, installer.exe and other Installer files.
5. Create the install.properties file for silent installation in the local folder.
6. With a batch file calling preinstall.exe and then querying the %errorlevel%, silent installation can be started and the exit code can be evaluated. See the sample batch file below.

Exit codes

0 = Success

1 = General Error in preinstall (e.g. not supported settings for user.language / user.country, for reason see preinstall_err.log)

2 = Unknown Error in preinstall

10 = General Error in Installer application (for reason see OL_Install_<timestamp>.log)

Sample batch file

```
@echo off
preinstall.exe

if errorlevel 10 goto err_installer
if errorlevel 2 goto err_unknown
if errorlevel 1 goto err_preinstall

echo Success
goto:eof

:err_installer
echo "Installer error - see OL_Install_<timestamp>.log"
goto:eof

:err_unknown
echo "Unknown preinstall error - see preinstall_err.log"
goto:eof

:err_preinstall
echo "Preinstall error - see preinstall_err.log"
goto:eof
```

Activating a License

PlanetPress Connect and PlanetPress Workflow 8 includes separate 30 day trial periods during which it is not necessary to have a license for reviewing basic functionality. If a modification to the license is required, such as to allow an extension to the trial period, or for extra functionality or plugins (e.g., the PReS Plugin for Workflow 8), then a new activation code will need to be requested.

Obtaining the PlanetPress Connect Magic Number

To obtain an activation file the OL™ Magic Number must first be retrieved. The Magic Number is a machine-specific code that is generated based on the computer's hardware and software using a top-secret Objectif Lune family recipe. Each physical computer or virtual computer should have a different Magic Number, thus require a separate license file to be functional.

To get the PlanetPress Connect Magic Number, open the PlanetPress Connect Designer application:

- Open the **Start Menu**
- Click on **All Programs**, then **Objectif Lune**, then **PlanetPress Connect**
- Open the **PlanetPress Connect Designer [version]** shortcut.
- When the application opens, if it has never been activated or the activation has expired, the **Software Activation** dialog appears:
 - **License Information** subsection:
 - **Magic Number**: Displays the PlanetPress Connect Magic Number.
 - **Copy to Clipboard**: Click to copy the Magic Number to the clipboard. It can then be pasted in the activation request email using the CTRL+V keyboard shortcut.
 - **Licensed Products** subsection:
 - **Name**: Displays the name of the application or module relevant to this activation.
 - **Serial Number**: Displays the activation serial number if the product has been activated in the past.
 - **Expiration Date**: Displays the date when the activation will expire (or the current date if the product is not activated)
 - **Web Activations**: Click to be taken to the online activation page (not yet functional).
 - **End-User License Agreement** (Appears only when loading a license file):
 - **License**: This box displays the EULA. Please note that this agreement is legally binding.
 - **I agree**: Select to accept the EULA. This option **must** be selected to install the license.
 - **I don't agree**: Select if you do not accept the EULA. You cannot install the license if this option is selected.
 - **Load License File**: Click to browse to the .olconnectlicense file, once it has been received.
 - **Install License**: Click to install the license and activate the software (only available when a license file is loaded).
 - **Close**: Click to cancel this dialog. If a license file has been loaded, it will not automatically be installed.

Note

The **Software Activation** dialog can also be reached through a shortcut located in **All Programs**, then **Objectif Lune**, then **PlanetPress Connect** and is named **Software Activation**. Since it does not load the software, it is faster to access for the initial activation.

Requesting a license

After getting the Magic Number, a license request must be done for both PlanetPress Connect and Workflow 8:

- **Customers** must submit their Magic Number and serial number to Objectif Lune via the Web Activations page: <http://www.objectiflune.com/activations>. The OL Customer Care team will then send the PlanetPress Connect license file via email.
- **Resellers** can create an evaluation license via the Objectif Lune Partner Portal by following the instructions there: <http://extranet.objectiflune.com/>

Note that if you do not have a serial number, one will be issued to you by the OL Activations team.

Accepting the license will activate it, after which the PlanetPress Connect services will need to be restarted. Note that in some case the service may not restart on its own. To resolve this issue, restart the computer, or start the service manually from the computer's Control Panel.

Activating PlanetPress Workflow 8

PlanetPress Workflow 8 uses the same licensing scheme as PlanetPress Connect. There are two ways of activating the license for Workflow 8 after saving it to a suitable location:

- If only PlanetPress Workflow 8 is installed, double-click on the license for the PlanetPress Workflow 8 License Activation dialog to open. Applying the license here activates all of the Workflow 8 components.
- If you have both PlanetPress Workflow 8 and PlanetPress Connect installed, it will not be possible to double-click on the license file as this will always open the PlanetPress Connect Activations Tool. Instead, open PlanetPress Workflow 8 manually and apply the license through the activations dialog within.

Activating PlanetPress Connect

To activate PlanetPress Connect, simply save the license file somewhere on your computer where you can easily find it, such as on your desktop. You can then load the license by double-clicking on it, or through the start menu:

- Open the **Start Menu**
- Click on **All Programs**, then **Objectif Lune**, then **PlanetPress Connect**
- Open the **PlanetPress Connect Designer [version]** shortcut. The “PlanetPress Connect Software Activation” tool displays information about the license and the [End-User License Agreement](#) (EULA).
- Click the **Load License File** button.
- Read the EULA and click I agree option to accept it.
- Click **Install License** to activate the license. The license will then be registered on the computer and you will be able to start using the software.

Warning

After installation message will appear warning that the Server services will need to be restarted. Just click OK to proceed.

Migrating to a new workstation

The purpose of this document is to provide a strategy for transferring a Connect installation to a new workstation. The following guide applies to OLConnect v1.x and Workflow v8.x.

Before installing the software

Before upgrading to a new version, even on a new workstation, consult the product's release note to find out about new features, bug fixes, system requirements, known issues and much more. Simply go to the [product page](#) and look for "Release notes" in the Downloads area.

You should also consult the following pages for some technical considerations before installing:

- [Network Considerations](#)
- [Database Considerations](#)

- [Environment Considerations](#)
- [Installation Pre-Requisites](#)
- Antivirus Exclusions

Downloading and Installing the Software

In order to migrate to a new workstation, the software must already be installed on the new workstation. Follow the [Installation and Activation Guide](#) to download and install the newest version of PlanetPress Connect on the new workstation.

Backing Up files from the current workstation

The first step in migrating to a new workstation would be to make sure all necessary production files and resources are backed up and copied over to the new system.

Technical

Although it is not necessary to convert all of your documents when upgrading to the latest version, we strongly recommended doing so. It is considered "Best Practice" to convert the documents to the version installed and then re-send them to the Workflow Tools.

Backing up Workflow files

To save all Workflow-related files, backup the entire working directory:

```
C:\ProgramData\Objectif Lune\PlanetPress Workflow 8
```

Here are a few important points when transferring these files:

- If you are upgrading to the latest version of Connect, it is recommended to open each template in Designer, produce a proof making sure the output is correct. Then send the template with its data mapper, job and output preset files to Workflow by clicking on **File - > Send to Workflow...**
- If you still use PlanetPress 7 legacy documents, PTK files can be imported by clicking on the Workflow tool button at the top left corner of the Workflow tool interface. If copying the `PlanetPress Workflow 8` folder directly, it's important to delete any file with the `.ps7` extension so as to refresh the postscript file for the new workstation.

- The Workflow configuration file itself is named ppwatch.cfg, and is backed up with the folders. However, it needs to be re-sent to the Service to be used. To do this, rename the file to .OL-Workflow, open the file with the Workflow tool, and send the configuration.
- Locate Custom Plugins (.dll) from the below folder on the old workstation and import them onto the new workstation
C:\Program Files (x86)\Common Files\Objectif Lune\PlanetPress
Workflow 8\Plugins
To import the plugins:
 - Start the Workflow Configuration Tool
 - Click on the Plug-in Bar
 - Click on the down pointing triangle under the **Uncategorized** group
 - Select **Import Plug-in** and select the .dll file.
- Import external scripts used by the **Run Script** plugin, making sure they reflect the same paths as on the previous workstation
- Install any external application, executable and configuration files used by the External Program plugin, making sure they reflect the same paths as on the previous workstation
- Reconfigure local ODBC connections. (i.e. create local copies of databases or recreate required DSN entries)
- Backup and import other custom configuration files, Microsoft Excel Lookup files, making sure they reflect the same paths as previously.
- Reinstall required external printer driver and recreate all Windows printer queues and TCPIP ports
- On the new workstation if the "TCP/IP Print Server" service is running in Windows, it is requested to disable that service so that it does not interfere with the Workflow LPD/LPR services.
- Configure the Workflow services account as in the previous installation. If accessing, reading and writing to network shares; it is recommended to use a domain user account and make it a member of the local Administrators group on the new workstation. Once the user account has been chosen:
 - Click on Tools in the Workflow Configuration menu bar
 - Click **Configure Services**
 - Select the user account
- If required, grant permissions to other machines (Designer clients and other servers) to send documents and jobs to the new server.

- Click on Tools in the Workflow Configuration menu bar
- Click on **Access Manager**
- Grant necessary permissions to remote machines
- Restart the Workflow Messenger service
- Reconfigure the Workflow Preferences as previously by clicking on the Workflow button on top left corner and clicking on Preferences:
 - Reconfigure the **Server Connection Settings** under **Behavior > OL Connect**
 - For PlanetPress Capture users, reconfigure the PlanetPress Capture options under **Behavior > PlanetPress Capture**
 - Reconfigure each of the plugin, where necessary, under **Plug-in** as previously. Capture OnTheGo users may want to enable the **Use PHP Arrays** option under **Plug-in > HTTP Server Input 1**
 - Send the configuration to local Workflow service

Backing up Connect Resources

The following resources are used by Connect and can be backed up from their respective folders:

- **Job Presets (.OL-jobpreset):**
C:\Users\<<UserName>\Connect\workspace\configurations\JobCreationConfig
- **Output Presets (.OL-outputpreset):**
C:\Users\<<UserName>\Connect\workspace\configurations\PrinterDefinitionConfig
- **OL Connect Print Manager Configuration files (.OL-ipdsprinter)**
C:\Users\<<UserName>\Connect\workspace\configurations\PrinterConfig
- **OL Printer Definition Files (.OL-printerdef)**
C:\Users\<<UserName>\Connect\workspace\configurations\PrinterDefinitionConfig
- **OMR Marks Configuration Files (.hcf)**
C:\Users\<<UserName>\Connect\workspace\configurations\HCFFiles

Other Resources

- **OL Connect Designer Templates**, DataMapper or Package files, copied from the folder where they reside.
- All Postscript, TrueType, Open Type and other **host based fonts** used in templates must be reinstalled on the new workstation.
- Import all **dynamic images** and make sure their paths match those in the old server.
- Make sure the new workstation can also access network or remote images, JavaScript, CSS, JSON, and HTML resources referenced in the Connect templates.

Secondary Software and Licenses

The following only apply for specific secondary products and licenses that interacts or is integrated into the main product.

Image, Fax and Search Modules

- Reconfigure the Image and Fax outputs with the new host information.
- Import the Search Profile and rebuild the database in order to generate the database structure required by the Workflow.

Capture

- Download the latest version of the [Anoto PenDirector](#).
- Before installing the PenDirector, make sure the pen's docking station isn't plugged into the server. Then install the PenDirector.
- Stop the Messenger 8 service on old and new server from the Workflow menu bar > Tools > Service Console > Messenger > right-click and select Stop.
- Import the following files and folders from the old server into their equivalent location on the new server:
C:\ProgramData\Objectif Lune\PlanetPress Workflow 8\PlanetPress Watch\capture\PPCaptureDefault.mdb
C:\ProgramData\Objectif Lune\PlanetPress Workflow 8\PlanetPress Watch\DocumentManager
C:\ProgramData\Objectif Lune\PlanetPress Workflow 8\PlanetPress Watch\PGC
- If Capture was previously using an external MySQL or Microsoft SQL Server, reconfigure the ODBC connection details as previously from the Workflow Preferences by clicking on the Workflow button on top left corner and clicking on Preferences, then reconfigure the

PlanetPress Capture options under Behavior >PlanetPress Capture > Use ODBC Database

- Start the Messenger 8 service on new server from the Workflow menu bar > Tools > Service Console > Messenger > right-click and select Start.

OL Connect Send

- Re-install OL Connect Send on the new Workstation. This should reinstall the OL Connect Send plugins in the Workflow Tool
- Reconfigure the Server URL and port during the OL Connect Send Printer Driver setup
- Re-run the OL Connect Send printer driver setup on client system and select the Repair option to point the clients to the new Server URL.

Configuring the Connect Engines

Any changes made to the Server preferences require the OLConnect_Server service to be restarted to take effect.

- Stop the OLConnect_Server service from Control Panel > Administrative Tools > Services > OLConnect_Server > Stop
- Configure the Merge and Weaver Engines scheduling preferences as in the previous installation
 - Open the Server Configuration from :
C:\Program Files\Objectif Lune\OL Connect\Connect Server\ServerConfig.exe
 - Configure the Merge and Weaver engines preferences under Scheduling (see Engine configuration)
 - Configure any other options for the Clean-up Service
- Configure the minimum (Xms) and maximum (Xmx) memory utilization for the Server, Merge and Weaver engines as previously or better (see Memory per engine):
 - Edit the following Xms and Xmx fields in the following configuration files:
 - C:\Program Files\Objectif Lune\OL Connect\Connect Server\Server.ini
 - C:\Program Files\Objectif Lune\OL Connect\MergeEngine\Mergeengine.ini

- C:\Program Files\Objectif Lune\OL Connect\weaverengine\weaverengine.ini

- Now start the **OLConnect_Server** service

Configuring the Server Extensions

In the case where the OLConnect MySQL is installed on the new Master Server, it is important to reconnect all Server Extension systems to the new Master Server.

Perform the following action on each Server Extension:

- Stop the OLConnect_ServerExtension service from **Control Panel > Administrative Tools > Services > OLConnect_ServerExtension > Stop**
- Open the Server Extension Configuration from:
C:\Program Files\Objectif Lune\OL Connect\Connect Server Extension\ServerExtension.exe
- Click on Database Connection and configure the JDBC Database connection settings so that the hostname points to the new Master server
- Click on Scheduling and type in the location of the new Master server
- Start the **OLConnect_ServerExtension** service

Transferring Software Licenses

Once all the above resources have been transferred over to the new server, it is recommended to thoroughly test the new system with sample files under normal production load to identify points of improvement and make sure the output match the user's expectation. Output generated at this point will normally bear a watermark which can be removed by transferring licenses from the old server to the new one.

- To transfer Connect and Workflow licenses, the user is usually required to complete a License Transfer Agreement which can be obtained from their [local Customer Care department](#)
- Upgrades cannot be activated using the automated Activation Manager. Contact your local Customer Care department.

To apply the license file received from the Activation Team:

- Start the PReS Connect, PlanetPress Connect or PrintShopMail Connect Software Activation module:
C:\Program Files\Objectif Lune\OL Connect\Connect Software Activation\SoftwareActivation.exe
- Click on Load License File to import the license.OLConnectLicense
- Start the Software Activation module on the Extension servers, where applicable
- Click on Load License File to import the above same license.OLConnectLicense
- Restart the OLConnect_Server service and restart the OLConnectServer_Extension service on the Extension servers, where applicable
- The number of Expected Remote Merge and Weaver engines should now be configurable in the Connect Server Configuration module (C:\Program Files\Objectif Lune\OL Connect\Connect Server Configuration\ServerConfig.exe)

To apply the PlanetPress Capture License

- Open the Workflow Configuration
- Click on Help on the Menu Bar and click on PlanetPress Capture License manager to import your license.

Uninstalling PlanetPress Connect from the previous workstation

It is recommended to keep the previous install for a few days until everything is completed. However, once your transition is successful and complete, the OL Connect software must be uninstalled from the original server.

Information about PlanetPress Workflow 8

If you wish to use PlanetPress Workflow (automation) in conjunction with PlanetPress Connect, you will need to install PlanetPress Workflow 8.8 onto the same machine. Workflow 8.8 is provided through a separate installer which is available on CD or for download as follows:

- If you are a **Customer**, the installer can be downloaded from the Objectif Lune Web Activations page: <http://www.objectiflune.com/activations>
- If you are a **Reseller**, the installer can be downloaded from the Objectif Lune Partner Portal: <http://extranet.objectiflune.com/>

PlanetPress Workflow 8 can be installed in parallel on the same machine as an existing PlanetPress® Suite 7.x installation.

Note however:

- If both versions need to be hosted on the same machine, PlanetPress Workflow 8.8 must always be installed after the legacy PlanetPress® Suite 7.x installation.
- When uninstalling PlanetPress Workflow 8.8, you may be prompted to repair your legacy PlanetPress® Suite 7.x installation.
- If PlanetPress Workflow 8.8 has been installed alongside PlanetPress® Suite 7, Capture can no longer be used with Workflow 7.
The plugins are now registered uniquely to Workflow 8.8 and the messenger for Workflow 7 is taken offline. It is only then possible to use Capture from PlanetPress Workflow 8.8.
- PlanetPress Workflow 8.8 and PlanetPress® Suite Workflow 7 cannot run simultaneously, since only one version of the Messenger service can run at a time. In fact, no two versions of Workflow can be run simultaneously on the same machine, regardless of versions.
- It is possible to switch between different versions running by shutting down one version's services and then starting the other. However, this is not recommended. There are no technical limitations that prevent processes from previous PlanetPress Suite Workflow versions (as far back as Version 4) to run on PlanetPress Workflow 8, removing the need to run both versions.

For more information on the licensing of Workflow 8.8, please see [Activating your license](#).

Upgrading from PlanetPress Suite 6/7

Note

This document is intended for people who already received their upgrade to PlanetPress Connect. They should already have their new serial number(s) in hand and the PlanetPress Connect installers.

With the release of PlanetPress Connect, Objectif Lune's innovative new technology, existing users of PlanetPress Suite version 7 and 6 have the possibility to migrate to an introductory version of PlanetPress Connect called "PlanetPress Connect Print-Only".

This migration benefits existing users in many ways and has limited impact on their current processes and how they use PlanetPress Suite version 7 and 6.

This document provides information on the migration process and the requirements and considerations for existing PlanetPress Suite users to upgrade to the latest generation of our products.

Note

PlanetPress Connect Print-Only is available for existing users of PlanetPress version 7 or 6 with a valid OL Care agreement. If you are using a previous version or are not covered by OL Care, please contact your reseller or your Objectif Lune Account Manager for more information.

What does PlanetPress Connect contain?

PlanetPress Connect is comprised of the following modules:

- PlanetPress Workflow 8. This is the natural evolution of PlanetPress Suite Workflow 7 (Watch, Office or Production). PlanetPress Workflow 8 is very similar to the PlanetPress Suite Workflow 7 version but contains some new features and has the ability to run PlanetPress Connect jobs, as well as PlanetPress Suite, PrintShop Mail Suite and PReS Classic documents.
 - Imaging for PlanetPress Connect is available as an option. It contains:
 - PlanetPress Fax
 - PlanetPress Image
 - PlanetPress Search
 - PlanetPress Capture is still supported in PlanetPress Workflow 8 but only with documents created with the PlanetPress Suite Design 7.
- PlanetPress Connect Designer. This is the design tool based on completely new technology. It is not backwards compatible and therefore cannot open PlanetPress Suite Design 7 documents. *If you want to continue editing those documents you can keep doing so in PlanetPress Suite Design 7.*
- PlanetPress Connect Server. This is the core of the Connect technology. This new module automates the merging of data with your new templates and generates the output. It is required for PlanetPress Workflow 8 to handle templates created with the PlanetPress Connect Designer. It can be installed on the same or a different machine as PlanetPress Workflow 8.

IMPORTANT: PlanetPress Connect does **not** contain the PlanetPress Design 7.

GOOD NEWS: PlanetPress Connect does not need any *printer licenses* to print from PlanetPress Connect or PlanetPress Suite. It can also print PrintShop Mail 7 and PReS Classic documents if these programs are licensed.

You can keep everything you have

The first thing to know is that you can keep your current PlanetPress Suite Workflow 7 configuration and your PlanetPress Suite Design documents. When upgrading to PlanetPress Connect, they will remain functional.

Please note that PlanetPress Suite Workflow 7 and PlanetPress Workflow 8 cannot run at the same time. See [Information about Connect Workflow 8](#) for information about these limitations. The only exception is the PlanetPress Suite Design tool that you can continue to use as it is not part of PlanetPress Connect.

For customers upgrading to the free “Print only” version, if you wish you to continue your OL Care engagement, the next year will be priced at the same price as your current price.

For customer upgrading to the full version of PlanetPress Connect, with or without new options, the next year of OL Care will be priced at the value of the new software you upgraded to.

Before going into any further details, please read the following section carefully.

PlanetPress Connect installation considerations

The PlanetPress Suite could run on a computer with a minimum of only 1GB of RAM available. The PlanetPress Connect Server with PlanetPress Workflow 8, by default, requires 8 GB of RAM, but if you intend on using the new PlanetPress Connect Designer on the same computer, you should consider having at least 12 GB of RAM available. See [System requirements](#).

Distributed installation or not

You can decide to install PlanetPress Connect modules all on the same computer or have each module on a different computer. Reasons for this could be:

- There is insufficient memory in the computer currently running PlanetPress Workflow 8 to also run PlanetPress Connect Server.

- You want to use a more powerful computer with more RAM and more cores to run the Server to achieve maximum performance.

What do I gain by upgrading to PlanetPress Connect?

PlanetPress Watch users

When upgrading to PlanetPress Connect, you receive key features of PlanetPress Office such as the following:

- Ability to input data from PDF
- Ability to print your PlanetPress Suite documents on any Windows printer (no need for printer licenses)
- Ability to create standard PDF output from your PlanetPress Suite documents
- Even if you don't recreate your existing PlanetPress Suite documents, you can easily change your workflow to convert your output to PDF, then output them in PCL to any device supporting it.

Note

If you were a PlanetPress Production user, you retain all functionalities within PlanetPress Workflow 8. These are automatically imported during the activation (see below).

Re-purpose your existing documents

IMPORTANT: PlanetPress Suite users covered by a valid OL Care contract receive a “Print only” version of PlanetPress Connect which can produce printed output. If you also own PlanetPress Imaging, which can produce PDF, Tiff and other archive formats, you will also receive a new version.

The full version of PlanetPress Connect can open your company to the digital world by enabling you to send HTML responsive emails as well as creating dynamic responses and interactive web pages. All that for a minimal fee. For more information on the full version of PlanetPress Connect, contact your reseller or your Objectif Lune Account Manager.

Upgrade to the full multi-channel version and expand onto the Web

If you choose to take the optional “multi-channel” upgrade, you can start right away to reuse the content of your existing documents and map it onto responsive documents that can be sent by email in full HTML glory and/or make them available as native HTML web pages using the latest CSS/JavaScript features.

IMPORTANT: If you owned them, you must also upgrade your Imaging modules to use the new PReS version.

Create new documents and integrate them into your workflow at your own pace

You can start benefiting from the innovative technology of the new PlanetPress Connect Designer right away by designing new documents, or re-doing existing ones at your own pace. With PlanetPress Connect Print-Only, you can now:

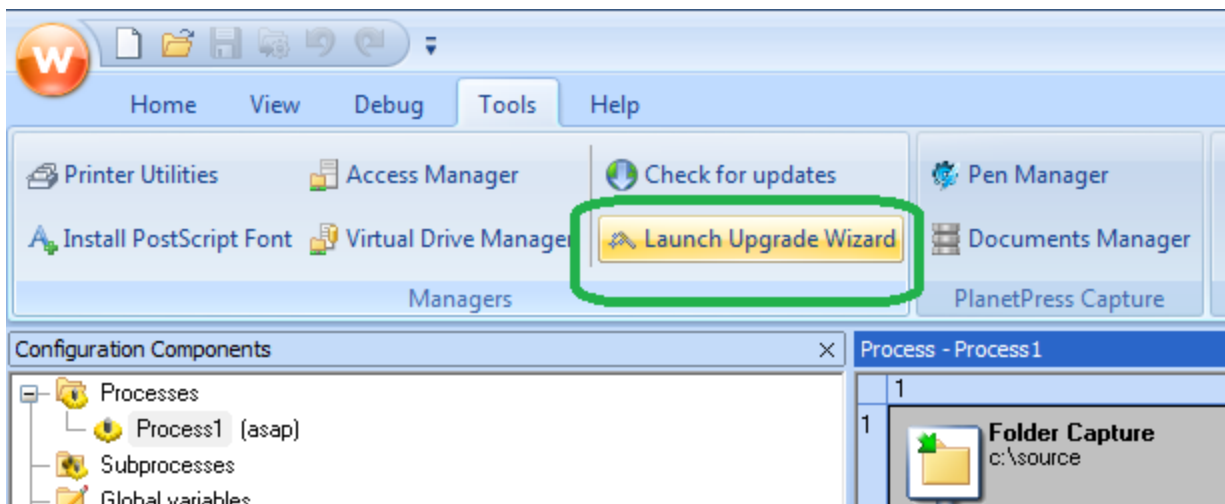
- Use the new Data Mapper to easily map any input data into a clean data model that any designer person can use
- Easily create documents with tables that spread over multiple print pages, respecting widow and orphan rules, displaying sub-totals and totals properly
- Have text that wrap around images

Upgrade steps

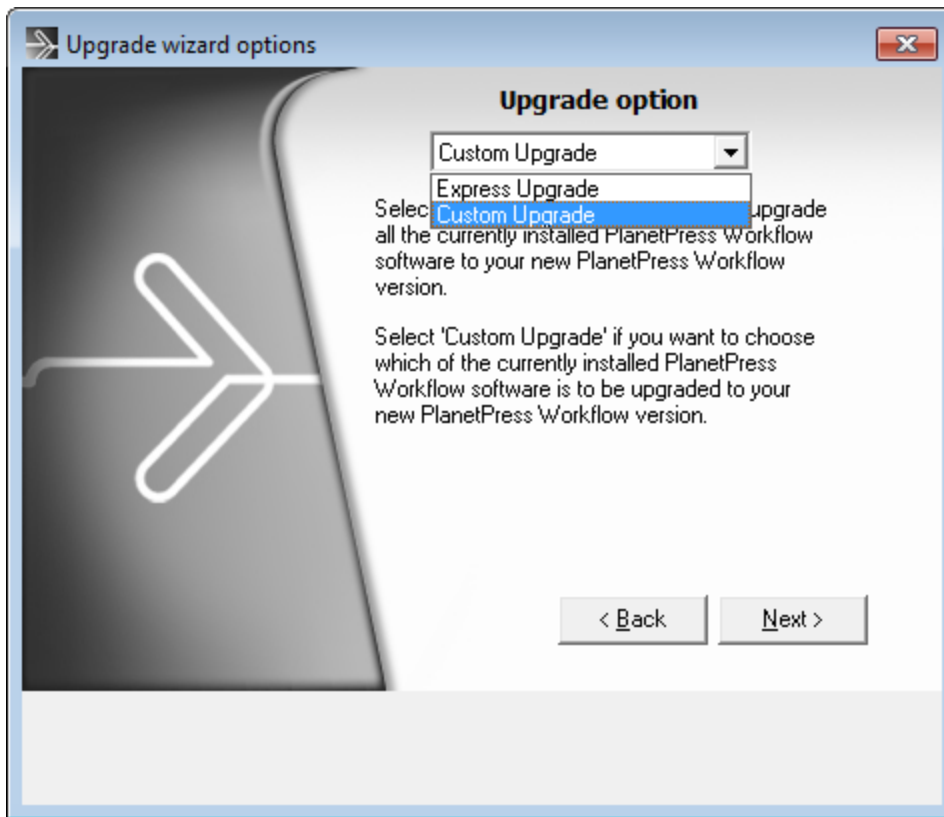
1. To upgrade to PlanetPress Connect, the first step is to stop your PlanetPress Workflow services. You can do so from the PlanetPress Workflow configuration tool or from the Windows Service Management console.
2. Then, using the PlanetPress Connect setup, install the Designer and/or Server on the appropriate computers. Then, using the PlanetPress Workflow 8 setup, install PlanetPress Workflow and/or PlanetPress Image on the appropriate computers. (See the installation and activation document for more details)
3. If you installed PlanetPress Workflow 8 on the same computer where you had PlanetPress Suite Workflow 6 or 7, you can use the Upgrade Wizard to import your:
 - PlanetPress Workflow:
 - Processes configuration
 - PlanetPress Suite compiled documents
 - Service configuration

- Access manager configuration
 - Custom plug-ins
- PlanetPress Fax settings
 - PlanetPress Image settings
 - PlanetPress Search profiles
 - Printer activation codes
 - PlanetPress Capture database
 - PlanetPress Capture pen licenses
 - Custom scripts
 - Content of your virtual drive
 - PlanetPress Messenger configuration
4. If you installed PlanetPress Workflow 8.8 on a different computer, please see "How to perform a Workflow migration" on page 70 for help importing all those settings, if you wish to import them.
 5. To launch the Upgrade wizard, open the PlanetPress Workflow 8 configuration tool and, from the Tools menu, launch the Upgrade Wizard.

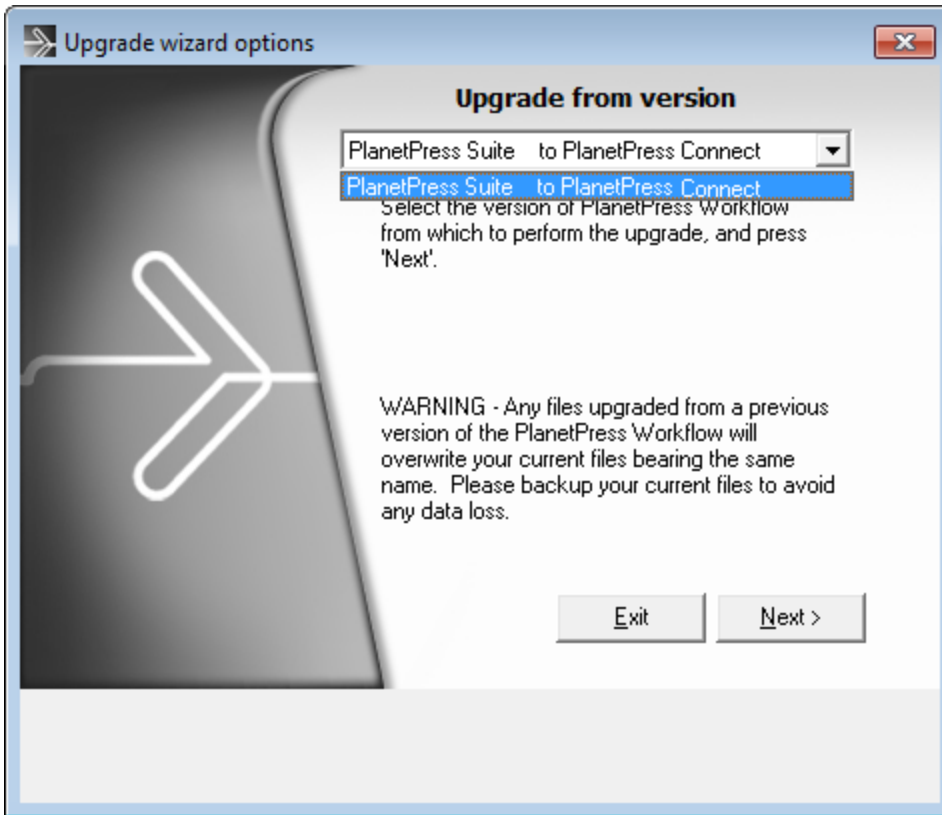
IMPORTANT: Before you start this process, make sure you have a backup of your current installation/computer.



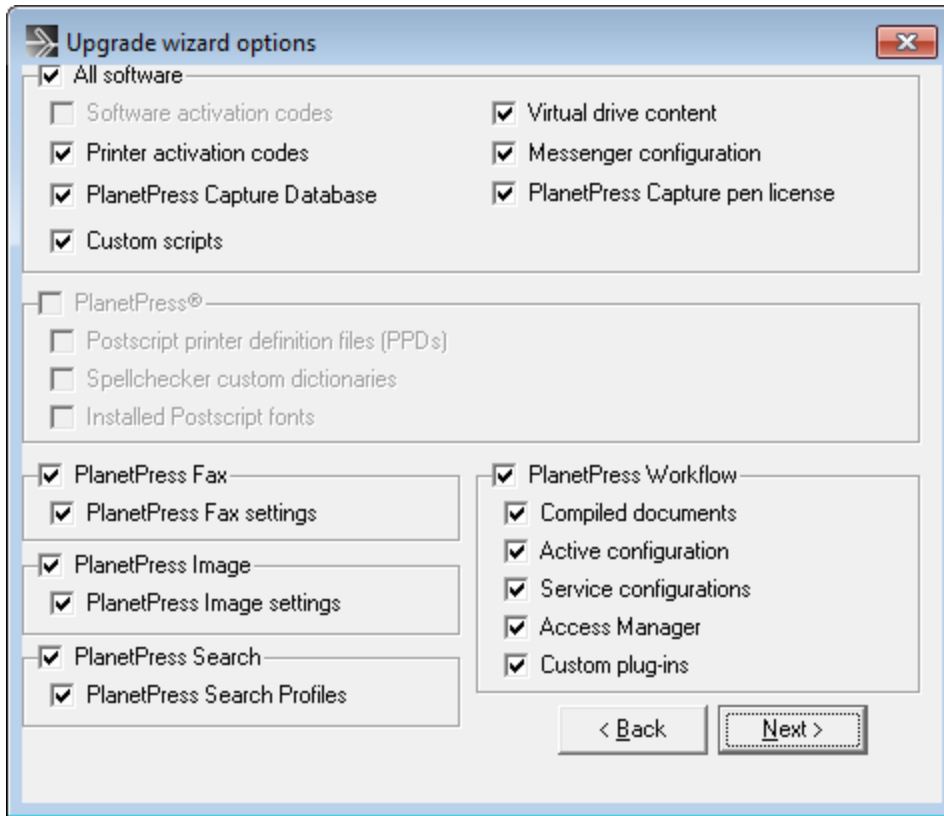
6. Then select your upgrade type:



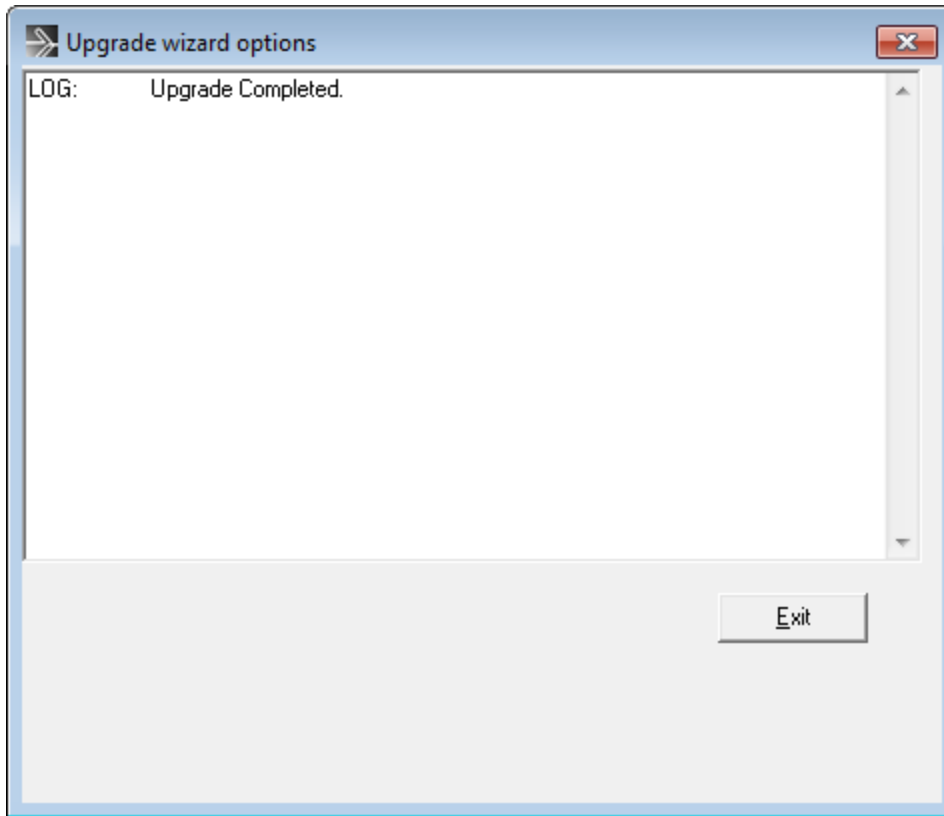
7. Then select the product from which you wish to upgrade:



8. If you selected to do a Custom upgrade, select the required options:



9. Then finally review the log in the final dialog for details on how it went:



10. After that you will need to get the activation file for your product.

To obtain your activation, download the PlanetPress Connect installer from the [Web Activation Manager](#), follow the instructions for the installation using the serial number provided to you. You can activate your license through the Web Activation Manager.

11. From now on, if you need to modify your PlanetPress Design documents, simply open PlanetPress Design 6 or 7, edit your document and send the updated version to PlanetPress Workflow 8. In order to do that:
 - If you have the PlanetPress Design on the same computer as the PlanetPress Workflow 8, you need to save the documents to PTK by using the "Send to" menu, then "PlanetPress Workflow", and there use the "Save to file" button. Then, from the PlanetPress Workflow 8 configuration tool, in the "Import" menu, select "Import a PlanetPress Document" and select the previously saved file.
 - If you have the PlanetPress Design on a computer and the PlanetPress Workflow 8 on another, you can simply use the "Send to" menu in the Designer and select the PlanetPress Workflow 8 to which you want to send the PlanetPress Design document.

How to perform a Workflow migration

What do you need to consider when upgrading from PlanetPress Suite 7 to PlanetPress Connect Workflow 8.8 on a new computer?

Installing and Activating Workflow 8.8 on a new computer

Points to consider:

- Before installing, be sure to read the [Installation and Activation Guide](#). There you will find detailed Connect Workflow installation steps as well as system requirements, notes on license activation and much more.
- It is recommended you retain your existing PlanetPress Suite installation for a period of time after the PlanetPress Connect Workflow 8.8 installation. We recommend this particularly when undertaking migration from one to the other. Once the migration has completed, you should uninstall PlanetPress Suite from your original installation. In the meantime, a fresh installation of PlanetPress Connect will run for 30 days without requiring an activation code, to simplify the migration process.
- Request new activation codes for your software licenses (License Transfer agreement needs to be filled out and signed). Contact your local Activations Department. www.objectiflune.com/activations
- Please note that PlanetPress Suite Workflow 7 and PlanetPress Workflow 8 cannot run at the same time. See [Information about Connect Workflow 8](#) for information about these limitations. The only exception is the PlanetPress Suite Design tool that you can continue to use as it is not part of PlanetPress Connect.

Printer Licences

If you are currently using Printer Licenses under PlanetPress Suite 7 and wish to continue doing so in PlanetPress Connect Workflow 8.8, there are a few ways in which you can reinstall those printer activation codes onto PlanetPress Connect Workflow 8.8. They are as follows:

- If you retained the .pac file (printer activation codes) from your previous installation, then double click on that file from within your new computer, and the printers will get activated.

If you did not retain the pac file, you can export a new printer activation code. This is done from the PlanetPress Suite Designer **Help > Printer Activation** menu option. When the "*Activate a printer*" dialog is launched, right click within it and select the *Export* context menu option, then save the file on the new computer. Double clicking on the .pac file will

then activate all of your printers on the new computer.

- Login to our Web Activation Manager (www.objectiflune.com/activations) using your customer number and password to get your Printer Activation Codes.
- If you do not have access to the computer in which PlanetPress Suite was previously installed, print a Status Page for each printer from your Connect Workflow 8 Configuration. Do this via the **Tools > Printer Utilities** menu option. Select "*Print Status Page*" and then select your printers from the list.

Email the Status Page(s) to activations@ca.objectiflune.com and you will receive a .pac file in return, with which you can activate your printer(s).

Documents and Resources

PlanetPress Suite Documents and Resources

- Backup all your PlanetPress Suite Design documents from your old computer and copy them onto the new computer. The files use the extension .ppX, where X is the version number of the PlanetPress Suite that created the files.
The documents do not have to be in any specific folder.
- Back up the entire directory of: "*C:\ProgramData\Objectif Lune\PlanetPress Suite 7\PlanetPress Watch\Documents*".
This folder contains all the PlanetPress Design documents and compiled forms (*.ptk and *.ptz).
Paste the files onto the new computer in the following folder:
"*C:\ProgramData\Objectif Lune\PlanetPress Workflow 8\PlanetPress Watch\Documents*"
- Back up the latest .pwX (PlanetPress Workflow Tools Configuration) file, found here: "*C:\ProgramData\Objectif Lune\PlanetPress Suite 7\PlanetPress Watch*".
Paste onto the new computer in the following folder:
"*C:\ProgramData\Objectif Lune\PlanetPress Workflow 8\PlanetPress Watch*"

There are several ways you can import Documents into PlanetPress Workflow. They are as follows.

1. In Connect Workflow go to **File > Import > PlanetPress Document ...** and select the .ptk document you wish to import.
These files will most likely be found in the Documents folder on the PlanetPress Suite computer:
"*C:\ProgramData\Objectif Lune\PlanetPress Suite 7\PlanetPress Watch\Documents*"

2. Copy all the PlanetPress Suite 7 Documents and Compiled forms (*.ptk and *.ptz) from the Documents folder on the PlanetPress Suite computer and paste them into the equivalent folder on the Connect Workflow Computer.
The PlanetPress Suite 7 folder would be "*C:\ProgramData\Objectif Lune\PlanetPress Suite 7\PlanetPress Watch\Documents*".
The PlanetPress Connect Workflow 8 folder will be "*C:\ProgramData\Objectif Lune\PlanetPress Workflow 8\PlanetPress Watch\Documents*".
3. Use the **File > Send To** menu option in PlanetPress Suite Designer and select the PlanetPress Connect Workflow 8 to which you want to send the PlanetPress Suite Designer document.
This should work with PlanetPress Suite versions 6 and 7.

Make sure that ports 5863 and 5864 are not blocked by firewall on either machine. Also make sure you add the PlanetPress Suite machine's IP address to the permissions list in Connect Workflow 8 from **Tools > Access Manager**. Further information about Workflow Access Manager can be found here: [Access Manager](#).

Windows Operating System Steps:

- Install all the Windows printer queues from the old computer, making sure they are named the same.
- If your existing documents referenced any local dynamic image resources in a folder or in Local Host, make sure that you import them onto the new computer as well, or make them available on a network accessible drive.
- Any special Postscript or TrueType fonts used will also need be installed on the new computer.
- Verify that you have access to any other resources that the PlanetPress Suite used. This includes network folders, printers, third party software and the like.

Workflow Plug-ins

Back up any custom PlanetPress Suite Workflow configuration Plug-ins (.dll) and copy them onto the new computer.

The PlanetPress Suite Workflow plug-ins folder can be found here:
"*C:\ProgramData\Objectif Lune\PlanetPress Suite 7\PlanetPress Watch\Plugins*".

Make sure that you copy only the custom plug-ins.


Alternatively, you can download custom plug-ins from this [link](#) onto the new computer.

Once you've copied your PlanetPress Suite Workflow configurations to Connect Workflow, you can confirm their availability through the Plug-in Bar **Uncategorized** category. There you will find all the Custom plug-ins that have been installed.

Missing plug-ins will be represented in Workflow steps through the use of a "?" icon. Such as in the following image, which shows that the "*TelescopingSortPlugin*" is not installed.



To import a plugin:

1. Click on the popup control () in the Plug-in bar.
2. Select **Import Plugin**
3. Browse to the location of the plug-in DLL file
4. Click on Open.
5. The new plug-in should appear in the Plug-in Bar **Uncategorized** category.

Configuring PlanetPress Connect Workflow 8

- Reconfigure any settings that may need to be applied to the PlanetPress Suite Messenger and PlanetPress Workflow Tools LPD services using the [Access Manager](#).
- All PostScript and TrueType host based fonts must be reinstalled. Make sure you restart the computer after this step.
- If necessary, reconfigure local ODBC connections. (i.e. create local copies of databases or recreate required DSN entries)
- Manually install all external executables that will be referenced by the Connect Workflow processes in the configuration file. If possible, retain the local path structure as used on the older installation.
- If the Windows "TCP/IP Print Server" service is running on the new computer, it is recommended that you disable the Server so that it does not interfere with the PlanetPress LPD/LPR services.

- If you are using images from a virtual drive, copy the entire contents of "C:\ProgramData\Objectif Lune\PlanetPress Suite 7\PSRIP" and paste them onto the new computer here: "C:\ProgramData\Objectif Lune\PlanetPress Workflow 8\PSRIP".
- Make sure to set the user who will run the PlanetPress Services. This is done by going into Tools/Configure services. The user will need to have local administration rights in order to be able to run the services.
For more information, see [Users and Configurations](#).
- Once all these steps have been completed, you will need to import your configuration file. Find the latest pwX file located on the old computer, if it is not already copied across to the new computer. The default location on the old computer is "C:\ProgramData\Objectif Lune\PlanetPress Suite 7\PlanetPress Watch\".

On the new computer you will need to **File > Import > Configuration Components**. Browse and find your file. If the file is not visible change the file type to *.pw7

PlanetPress Image, Fax and Search

- Reconfigure the PlanetPress Image and PlanetPress Fax outputs with the new host information.
- You must import the Search Profile and rebuild the database in order to generate the required database structure.

PlanetPress Capture

- If you have a Capture Solution, please see "How to perform a Capture migration" below.

How to perform a Capture migration

This page provides information on how to conduct a proper migration of a [Capture](#) solution.

These steps must be executed **after** a proper Workflow Migration has been completed. Instructions on how to do such can be found here: "How to perform a Workflow migration" on page 70.


Failure to do so will result in unexpected problems.

Note

It is recommended that you first update your PlanetPress Suite to version 7.6 before cross-grading to PlanetPress Connect.

Using PlanetPress Connect Workflow 8.8 on the same computer as PlanetPress Suite 7.6

Steps to migrate:

1. Update existing installation to PlanetPress Suite version 7.6 if not already done.
2. Install PlanetPress Connect Workflow 8.8 on the same computer.
3. Do the following for **both** PlanetPress Suite version 7.6 and PlanetPress Connect Workflow 8.
 1. Open Workflow **Service Console**. This can be done either via the Windows Start Menu, or from within Workflow Configuration application (via menu option **Tools > Service Console**).
 2. Select **Messenger** in the tree list, right click and select  **Stop** from the context menu.


Note

These steps must be done for **both** PlanetPress Suite Workflow 7 and PlanetPress Connect Workflow 8.

4. Copy the file **PPCaptureDefault.mdb** from this folder:
`"C:\ProgramData\Objectif Lune\PlanetPress Suite 7\PlanetPress Watch\capture"`
to this folder:
`"C:\ProgramData\Objectif Lune\PlanetPress Workflow 8\PlanetPress Watch\capture"` and overwrite the existing database.

Note

Prior to PlanetPress Suite 7.6, all Capture patterns, documents and several other details were contained within the one single database. As of PlanetPress Suite 7.6 a separate database has been used for the patterns alone (**PPCaptureDefault.mdb**).

5. Copy the contents of this folder:
"C:\ProgramData\Objectif Lune\PlanetPress Suite 7\PlanetPress Watch\DocumentManager"
to this folder:
"C:\ProgramData\Objectif Lune\PlanetPress Workflow 8\PlanetPress Watch\DocumentManager".
6. Copy the contents of this folder:
"C:\ProgramData\Objectif Lune\PlanetPress Suite 7\PlanetPress Watch\PGC"
to this folder:
"C:\ProgramData\Objectif Lune\PlanetPress Workflow 8\PlanetPress Watch\PGC"
7. Restart the PlanetPress Connect Workflow 8 Messenger. To do this,
 1. Open Workflow **Service Console**. This can be done either via the Windows Start Menu, or from within Workflow Configuration application (via menu option **Tools > Service Console**).
 2. Select **Messenger** in the tree list, right click and select  **Start** from the context menu options.
8. Contact your local Objectif Lune activation team and transfer any Pen(s) licenses across.

Using PlanetPress Connect Workflow 8.8 on a different computer to PlanetPress Suite 7.6

Tip

It is safer to migrate outside high peak production since the Capture solution cannot be run in parallel on two computers.

Once the Capture database has been transferred to the new computer, any update made to the old computer will be lost unless the steps to migrate are reproduced again.

Once a Pen has been docked and the data transfer done, its memory is wiped, thus rendering the parallel mode very hard to produce. It is not impossible, but describing how it can be done this is beyond the scope of this migration article.

Steps to migrate:

1. Update existing installation to PlanetPress Suite version 7.6 if not already done.
2. Install PlanetPress Connect Workflow 8.8 on new computer.
3. The **Anoto PenDirector** must be installed. If it is not, you can download it from [here](#) and then install it.

Note


It is strongly recommended that you install the latest version of the PenDirector. Please use the link provided on the previous line.

Do not get any other version of the PenDirector from the Anoto website, as they will not have been set up correctly for our Capture solution.

Note

Prior to installation, make sure you unplug the Pen docking station from the USB port on the computer where you are about to install the Anoto PenDirector.

4. Do the following for **both** PlanetPress Suite version 7.6 and PlanetPress Connect Workflow 8.
 1. Open Workflow **Service Console**. This can be done either via the Windows Start Menu, or from within Workflow Configuration application (via menu option **Tools > Service Console**).

2. Select **Messenger** in the tree list, right click and select  **Stop** from the context menu.

Note


These steps must be done for **both** PlanetPress Suite Workflow 7 and PlanetPress Connect Workflow 8.

5. Copy the file **PPCaptureDefault.mdb** from this folder on the PlanetPress Suite 7.6 computer:
"C:\ProgramData\Objectif Lune\PlanetPress Suite 7\PlanetPress Watch\capture"
to this folder on the new PlanetPress Connect Workflow 8.8 computer:
"C:\ProgramData\Objectif Lune\PlanetPress Workflow 8\PlanetPress Watch\capture" and overwrite the existing database.

Note

Prior to PlanetPress Suite 7.6, all Capture patterns, documents and several other details were contained within the one single database. As of PlanetPress Suite 7.6 a separate database has been used for the patterns alone (**PPCaptureDefault.mdb**).

6. Copy the contents of this folder on the PlanetPress Suite 7.6 computer:
"C:\ProgramData\Objectif Lune\PlanetPress Suite 7\PlanetPress Watch\DocumentManager"
to this folder on the new PlanetPress Connect Workflow 8.8 computer:
"C:\ProgramData\Objectif Lune\PlanetPress Workflow 8\PlanetPress Watch\DocumentManager".
7. Copy the contents of this folder on the PlanetPress Suite 7.6 computer:
"C:\ProgramData\Objectif Lune\PlanetPress Suite 7\PlanetPress Watch\PGC"
to this folder on the new PlanetPress Connect Workflow 8.8 computer:
"C:\ProgramData\Objectif Lune\PlanetPress Workflow 8\PlanetPress Watch\PGC"
8. Restart the PlanetPress Connect Workflow 8 Messenger. To do this,
 1. Open Workflow **Service Console**. This can be done either via the Windows Start Menu, or from within Workflow Configuration application (via menu option **Tools**

- > **Service Console**).
- 2. Select **Messenger** in the tree list, right click and select  **Start** from the context menu options.
- 9. Contact your local Objectif Lune activation team and transfer any Pen(s) licenses across.

Known Issues

This page lists important information that applies to PlanetPress Connect 1.8.

Issues with Microsoft Edge browser

The Microsoft Edge browser fails to display web pages when the Workflow's CORS option (in the HTTP Server Input 2 section) is set to "*". This issue will be resolved in a future release.

Workflow - "Execute Data Mapping" - Issues with multiple PDFs

If a process has a *Folder Capture* step (but not the first input step) to capture multiple PDFs within a folder, followed by a *Execute Data Mapping* step to extract XML files for each PDF, only the first PDF file will be processed. The workaround is to put the Extract Datamapper in a Branch, just after the Folder Input. This issue will be fixed in a later release. (SHARED-59752)

Installation Paths with Multi-Byte Characters

When installing the Chinese (Traditional or Simplified) or Japanese versions of Connect, if the user specifies an alternative installation path containing multi-byte/wide-char characters it can break some of the links to the Connect-related shortcuts in the Start Menu and cause an error to appear at the end of the installer. The workaround for the moment is to use the default installation path. The problem will be addressed in a later release.

Switching Languages

Changing the language using the **Window>Preferences>Language Setting** menu option does not currently change all of the strings in the application to the selected language. This is a known issue and will be fixed in a later release.

In the meantime we offer the following workaround for anyone who needs to change the language:

1. Go to the .ini files for the Designer and Server Config:
 - C:\Program Files\Objectif Lune\OL Connect\Connect Designer\Designer.ini
 - C:\Program Files\Objectif Lune\OL Connect\Connect Server Configuration\ServerConfig.ini
2. Change the language parameter to the required one under Duser.language=en | es | de | fr | it | ja | ko | pt | tw | zh

Only one of the above language tags should be selected. Once saved, Connect will appear in the selected language at next start-up.

GoDaddy Certificates

When installing Connect offline, dialogs allow installing the GoDaddy certificates. Most users should use the default settings and click **Next**. In some cases, however, this may not work correctly. For this reason those users should activate **Place all certificates in the following store** and then select the **Trusted Root Certification Authorities** as the target certificate store.

MySQL Compatibility

After installing Connect 1.8 a downgrade to a Connect version earlier than Connect 1.3 or to a MySQL version earlier than 5.6.25 is not seamlessly possible. This is because the database model used in Connect 1.3 and later (MySQL 5.6) is different to that used in earlier versions. If you need to switch to an older version of Connect / MySQL, it is first necessary to remove the Connect MySQL Database folder from "%ProgramData%\Connect\MySQL\data" before installing the older version.

PostScript Print Presets


The print presets for PostScript were changed from Version 1.1 onwards meaning that some presets created in Version 1.0 or 1.0.1 may no longer work.

Any PostScript print preset from Version 1.0 that contains the following will not work in Version 1.8: *.all[0].*

Any preset containing this code will need to be recreated in Version 1.8.

Available Printer Models

Note that only the single Printer Model (Generic PDF) will appear on the **Advanced** page of the **Print Wizard** by default.

To add additional printer models click on the settings  button next to the Model selection entry box.

Note that the descriptions of some of the printers were updated in version 1.2 meaning that if you had version 1.n installed, you may find that the same printer style appears twice in the list, but with slightly different descriptions.

For example the following printer types are actually identical:

- Generic PS LEVEL2 (DSC compliant)
- Generic PS LEVEL2 (DSC)

External Resources in Connect

There are certain limitations on how external resources can be used in Connect. For example if you want to link a file (e.g., CSS, image, JavaScript etc.) from a location on the network but you do not want to have a copy of the file saved with the template you need to do the following:

1. The resource must be located where it can be accessed by all Servers/Slaves run as users. Failure to do this will cause the image to appear as a Red X in the output for all documents which were merged by engines which could not access the file. The job will terminate normally and the error will be logged.
2. The file must be referenced via a UNC path e.g.,
file:///w2k8r2envan/z%20images/Picture/Supported/JPG/AB004763.jpg
 - UNC paths are required because the services will be unable to access mapped network drives (Windows security feature).
 - The engine processing the job will look on the local file system for the direct file path leading to the “resource not found” issue mentioned above.

Warning

Important Note: The Designer itself and Proof Print do not use processes that run as

services and they may find local files with non-UNC paths which can lead to the false impression that the resources are correct.

Using Capture After Installing Workflow 8

If PlanetPress Connect Workflow 8 is installed alongside PlanetPress Suite Workflow 7, Capture can no longer be used within Workflow 7. The plugins are now registered uniquely to Workflow 8 and the Messenger for Workflow 7 is taken offline. It is only possible to use Capture from PlanetPress Connect Workflow 8 thereafter.

Capturing Spool Files After Installing Workflow 8

If PlanetPress Connect Workflow 8 is installed alongside PlanetPress Suite Workflow 7, the PlanetPress Suite 7 option to capture spool files from printer queues will no longer function. The solution is to use PlanetPress Connect Workflow 8 to capture spool files from printer queues.

Colour Model in Stylesheets

The colour model of colours defined in a stylesheet can sometimes change after editing the stylesheet. This is a known issue and will be addressed in a subsequent release.

Image Preview in Designer

If in the Windows Internet settings (**Connection Settings > LAN configuration**) a proxy is enabled, but "Bypass proxy settings for local addresses" is not checked, the image preview service, conversion service and live preview tab in the Designer will not work and exhibit the following issues:

- Images will be shown as 0 size boxes (no red 'X' is displayed)
- Live preview does not progress, and when re-activated reports "browsers is busy"

To fix the issue you must check the "Bypass proxy settings for local addresses" option

Merge\Weaver Engines when Printing

The print operation in the Designer will automatically detect whether the Merge\Weaver engines are available and display a message for the user to retry or cancel if not. Once the Merge\Weaver engine becomes available and the user presses retry the print operation will proceed as normal. This message can also occur in the following circumstances:

- If the server is offline and you are not using Proof Print
- On some occasions before the Print Wizard opens

REST Calls for Remote Services

The Server will now accept REST calls for all remote services and will make commands wait indefinitely until the required engines become available. The Server will log when it is waiting for an engine and when it becomes available. Note that there is no way to cancel any commands other than stopping the Server.

Print Content and Email Content in PlanetPress Workflow

In PlanetPress Workflow's Print Content and Email Content tasks, the option to Update Records from Metadata will only work for fields whose data type is set to String in the data model. Fields of other types will not be updated in the database and no error will be raised. This will be fixed in a later release.

Print Limitations when the Output Server is located on a different machine

The following limitation may occur when using the Print options from a Designer located on a different machine to the Output Server:

- The file path for the prompt and directory output modes is evaluated on both the client AND server side. When printing to a network share it must be available to BOTH the Designer and Server for the job to terminate successfully.
- The Windows printer must be installed on both the Server and Designer machines.
- When printing via the Server from a remote Designer, the output file remains on the Server machine. This is remedied by selecting "Output Local" in the Output Creation configuration

VIPP Output

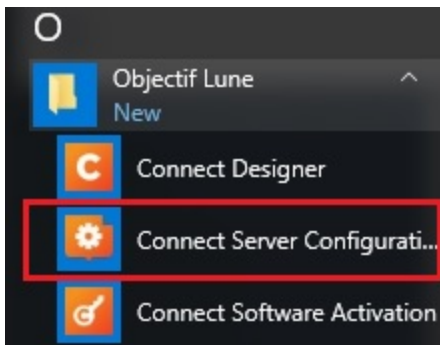
Some templates set up with landscape orientation are being produced as portrait in VIPP. It can also sometimes be the case that text and images can be slightly displaced. These are known issues and will be addressed in a later release of Connect.

Server Configuration Settings

This chapter describes configuring the PlanetPress Connect Server.

The Connect Server settings are maintained by the **"Connect Server Configuration"** utility tool which is installed alongside PlanetPress Connect.

"Connect Server Configuration" can be launched from the Start Menu, as seen in the following screen-shot:



The **"Connect Server Configuration"** dialog is separated into individual pages, where each page controls certain aspects of the software.

The following pages are available:

- "Clean-up Service preferences" on page 697
- "Database Connection preferences" on page 700
- "Language Setting Preferences" on page 710
- ["Scheduling Preferences" on the next page](#)
 - "Merge Engine Scheduling" on page 86
 - "Weaver Engine Scheduling" on page 88
- "Server Security Settings" on page 90

Scheduling Preferences

The scheduling preferences are a way to control precisely how the PlanetPress Connect services work in the background.

Whenever an operation is scheduled:

- The number of instances to use is determined based on whether the operation is small, medium or large
- If there is a reserved instance for that type of command available then it will use a reserved instance
- If no reserved instances are available then any unreserved instance that is available will be used
- If no instances are available then the command will be blocked until an appropriate instance becomes available

Technical

For more information on instances and performance, see [Performance Considerations](#).

Scheduling Properties

- **Definitions group:** Defines what is to be considered a **Small** or **Large** job. Anything in between is considered a **Medium** job.
 - **Maximum records in a small job:** Enter the maximum number of records for a job to be considered **Small**.
 - **Minimum records in a large job:** Enter the minimum number of records for a job to be considered **Large**.

Note

Changes made to these settings will be applied on the run. Existing jobs will be taken into account when determining if a job can run.

For instance, if the minimum records for a Large job is increased from 1,000 to

10,000 and a job of 2,000 records is already running, then this existing job will now be considered a Medium job.

Likewise, if the minimum records for a Large job is decreased from 10,000 to 5,000 and a job of 7,000 records is already running, then this existing job will now be considered a Large job.

Additional Scheduling Preferences:

Scheduling Preferences can be applied to the two distinct Engines used in the Connect production process. The **Merge Engine** merges the template and the data to create Email and Web output, or to create an intermediary file for Printed output. The intermediary file is in turn used by the **Weaver Engine** to prepare the Print output.

Each of these Engines has its own Scheduling Preference page:

- "Merge Engine Scheduling" below
- "Weaver Engine Scheduling" on page 88

Merge Engine Scheduling

The **Merge Engine** merges the template and the data to create Email and Web output, or to create an intermediary file for Printed output. The intermediary file is in turn used by the **Weaver Engine** to prepare the Print output.

This preferences page defines how different instances and speed units are attributed to different jobs when creating Content Items (Print Content, as well as Web and Email output Content generation). For information on the terminology and some performance tips, see [Performance Considerations](#). Note that *in this dialog*, the use of the word "Engine" is synonymous to both "Instance" and "Speed Unit" since a single Speed Unit can be used for each Engine.

Email and Web output is generated *only* with the Merge Engine and thus their output speed is limited through this engine. However, the output speed of Print jobs is limited through the Weaver Engine, so when Print Content is generated through the Merge Engine, its speed is not

limited. Additionally, you may launch up to 256 engines for Print Content generation, but Email and Web may only use the number of engines permitted by your license.

Note

Changes made to the following settings will be applied on the run (when the Apply button is pressed), and do not require the OLCConnect_Server service to be restarted.

- **Use one internal engine:** Check to limit to a single instance of the server. Useful for computers that run below the recommended [System requirements](#), or demo machines.
- **Total Engines Available:** Read-only box indicating the current number of engines that are active or available.
- **Engines Launched:** Enter the total number of Merge Engines desired on this server. When changing the number of engines, it is necessary to save this dialog (Apply) to actually apply the changes.
- **Reserved Count:** Read-only box indicating the total number of "Reserved" engines, as set in the Speed unit reservations area below.
- **Restart After (mins):** Due to a currently un-fixable memory leak in some libraries used by PlanetPress Connect, it is necessary to restart our engines after a certain amount of time. The default is generally sufficient for all our clients. Only change on the advice of a technical support agent.
- **Parallel Engines/Speed Units per job:** This area determines how many engines and speed units are used for *each* job that runs through it. In short, if any specific type of job has more than one parallel speed unit assigned to it, that many engines will be used to run each of its jobs. This is in fact a *multiplier of speed*, however it is a *divider of the number of jobs* that can be run simultaneously. You cannot attribute more parallel speed units than what you have available for any specific type of job, and you require at least that number of floating speed units, or reservations, for the required parallel speed units required.
- **Engine reservations:** This area is used to reserve engines specifically for certain types of jobs. Reserved engines *cannot be used* by any other type of job.
 - **Floating:** Read-only box indicating the number of floating engines that can be used for any type of job. This number is equal to the **Total Engines Available - Reservations**. For example if 6 engines are launched and 4 are reserved for small jobs, 2 will be Floating.

- **Small job speed unit reservations:** Enter a number of engines reserved for small print jobs.
- **Medium job speed unit reservations:** Enter a number of engines reserved for medium print jobs.
- **Large job speed unit reservations:** Enter a number of engines reserved for large print jobs.
- **Email engine reservations:** Enter a number of engines reserved for Email jobs.
- **HTML engine reservations:** Enter a number of engines reserved for Web jobs.
- **Maximum concurrent engine per type:** This area defines the maximum possible Engines used for any specific job type. The limit needs to be at least the number of reservations or parallel speed units, whichever is lowest.
 - **Small print job limit:** Enter the maximum number of engines that can run small print jobs.
 - **Medium print job limit:** Enter the maximum number of engines that can run medium print jobs.
 - **Large print job limit:** Enter the maximum number of engines that can run large print jobs.
 - **Email limit:** Enter the maximum number of engines that can run Email jobs.
 - **Maximum Email limit in license:** Read-only box indicating the maximum number of engines useable for Email content creation.
 - **HTML limit:** Enter the maximum number of engines that can run Web jobs.
 - **Maximum HTML limit in license:** Read-only box indicating the maximum number of engines useable for Web content creation.

Weaver Engine Scheduling

The **Merge Engine** merges the template and the data to create Email and Web output, or to create an intermediary file for Printed output. The intermediary file is in turn used by the **Weaver Engine** to prepare the Print output.

This preference page determines the number of Weaver engines launched, as well as their speed, when generating the output. One single engine can only process a single job at a time, at the speed available depending on licence and configuration. Note that *in this dialog*, the use of the term "Speed Unit" is in relation of the available speed for each engine. One speed unit = one unit of speed at the maximum speed your licence and number of Performance Packs

allows. With no Performance Pack, PlanetPress Connect's Weaver engine can generate output at 500 ppm (pages per minute). Additional Performance Packs increase this base speed per engine.

Changes made to the following settings will be applied on the run (when the Apply button is pressed), and do not require the OLConnect_Server service to be restarted.

- **Use one internal engine:** Check to limit to a single instance of the server. Useful for computers that run below the recommended system requirements (see "System Requirements" on page 27) or demo machines.
- **Total Engines Available:** Read-only box indicating the current number of engines that are active or available.
- **Local Engines Launched:** Enter the total number of Weaver Engines desired on this server. When changing the number of engines, it is necessary to save this dialog (Apply) to actually apply the changes.
- **Speed Units Launched:** Read-only box indicating the number of speed units launched.
- **Limit in license:** Read-only box indicating the maximum number of speed units useable to produce output.
- **Reserved Count:** Read-only box indicating the total number of "Reserved" engines, as set in the **Speed unit reservations** topic below.
- **Restart after (mins):** Restarts the Weaver Engines at the selected time interval. If a job is in progress at that point, the Weaver Engines will await job completion before restarting.
- **Parallel engines/speed units per job:**
 - **Parallel engines/speed units per medium job:** Enter the number of engines/speed units that prioritize medium print jobs.
 - **Parallel engines/speed units per large job:** Enter the number of engines/speed units that prioritize large print jobs.
- **Speed unit reservations:**
 - **Floating:** Read-only box indicating the number of floating speed units that can be used for any type of job.
 - **Small job speed unit reservations:** Enter a number of speed units reserved for small print jobs.
 - **Medium job speed unit reservations:** Enter a number of speed units reserved for medium print jobs.

- **Large job speed unit reservations:** Enter a number of speed units reserved for large print jobs.
- **Maximum speed units:**
 - **Small job limit:** Enter the maximum number of speed units that can run small print jobs.
 - **Medium job limit:** Enter the maximum number of speed units that can run medium print jobs.
 - **Large job limit:** Enter the maximum number of speed units that can run large print jobs.

Server Security Settings

This dialog controls the security settings for external applications connecting to the PlanetPress Connect Server, such as PlanetPress Workflow or scripts communicating through the REST API.

Warning

It is **highly recommended** to keep security enabled and change the password on any server that accessible from the web. If these precautions are not taken, data saved in the server may be accessible from the outside!

- **Enable server security:** Enable to add authentication to the REST server.
 - When enabled, the **same** username and password (which cannot be blank) must be entered in any remote Connect Designer that links to this Server. The Designer username and password entries can be found under the "Print Preferences" on page 710 sub-section of the Designer Preferences dialog.
 - When disabled, a username and password is not required to make REST request, and tasks in PlanetPress Workflow do not require them in the Proxy tab. Nor would a username and password be required on any remote Connect Designer that links to this Server.
- **Administrator's username:** Enter the username for the server security. The default username is **ol-admin**.
- **Administrator's password:** Enter a password for the server security. The default password is **secret**.

- **Confirm password:** Re-enter the password for the server security.
- **Default session length (min):** Enter a session time (in minutes) that the authentication stays valid for the requested process. This can reduce the number of requests to the server since an authentication request is not necessary during the session.

Uninstalling

This topic provides some important information about uninstalling (removing) PlanetPress Connect1.8.

To uninstall PlanetPress Connect select the application from within the Add/Remove programs option under the Control Panel. This will start the **PlanetPress Connect Setup Wizard** in uninstall mode.

Note

The **PlanetPress Connect Setup Wizard** might take some seconds to appear.

Important Note: Stop any active Anti-Virus software before uninstalling Connect.

Some anti-virus systems are known to block the uninstallation of MySQL datafiles, as well as blocking the uninstallation of the MySQL database application itself. Therefore it is **highly recommended** that any anti-virus application be stopped prior to uninstalling PlanetPress Connect, as otherwise the Connect uninstallation might not work correctly.

Impacts upon other Applications and Services

- The Uninstall will terminate the installed Server / MySQL service(s)
- The following applications / services should be stopped in a controlled fashion, before running the PlanetPress Connect Uninstall:
 1. PlanetPress Connect
 2. Connect products on remote systems which refer to this MySQL database.

3. Any Connect Workflow using PlanetPress Connect plugins which connect to this server.

Uninstallation Wizard

The uninstallation is done by running the PlanetPress Connect Setup Wizard in uninstall mode. The Wizard consists of the following pages:

1. **PlanetPress Connect Setup:** An information page, listing what will be uninstalled, and also warning about impacts upon running Applications and Services.
2. **Data Management:** A page that provides options for backing up or deleting Connect data. Selections are as follows:
 - **Delete Connect Workspace Data:** Check this box to delete the Workspace data for the current user, or for selected users (as determined by the "Select Users" button)
 - **Backup Connect Workspace Data for all specified Users:** Check this box to backup the Workspace data for the specified users (as previously determined) into a compressed ZIP file (whose location can be customized), before deletion of the full Workspace data.
 - **Delete MySQL objectlune Data:** Check this box to delete the MySQL database installed with PlanetPress Connect.
 - **Backup MySQL Date:** If the deletion check box is selected, this option appears to allow backing up the MySQL database to a customizable location, prior to uninstallation.

General information

Connect consists of visible and invisible parts. The invisible parts process the Connect job to provide the actual output. They are introduced to you in the topic: "Connect: a peek under the hood" below.

For a list of all file types used in Connect, see: "Connect File Types" on page 98.

You can find additional information that complements the user manuals, such as error codes and frequently asked questions about PlanetPress Connect, in the [Knowledge base](#).

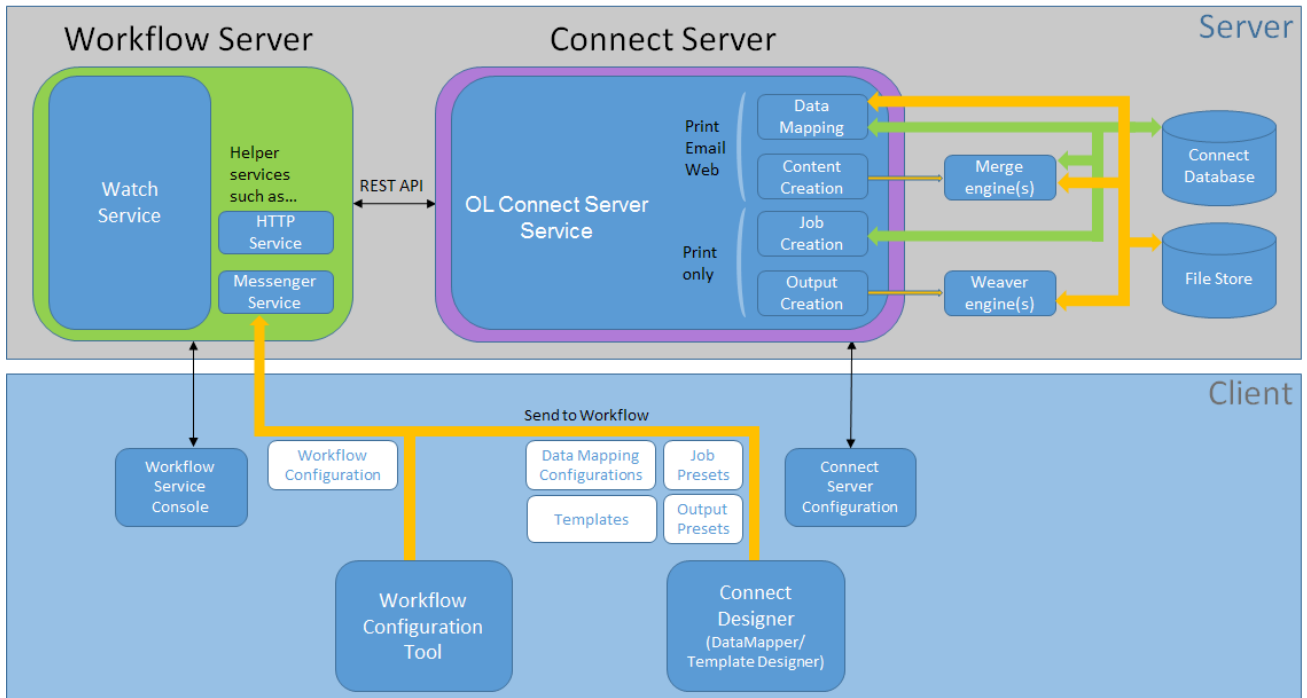
Connect: a peek under the hood

Connect consists of visible and invisible parts.

The visible parts are the tools that you use to create templates, data mapping configurations, and print presets (the Designer/DataMapper), and to create Workflow configurations (the Workflow configuration tool).

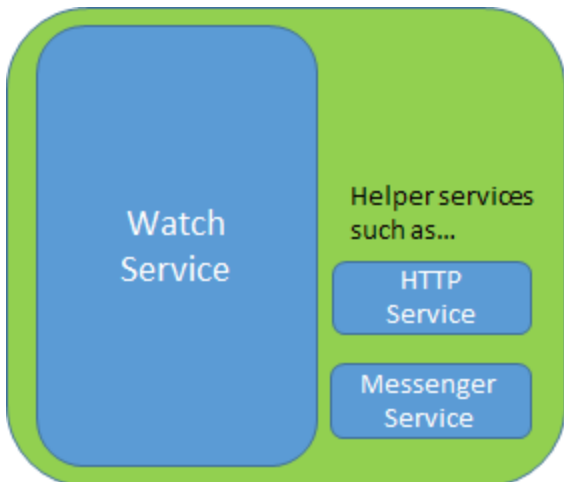
The invisible parts process the Connect job to provide the actual output. This topic introduces you to those parts.

Here's a simplified, graphical representation of the architecture of PlanetPress Connect. The components described below are all located in the 'Server' part.



The Workflow server

The Workflow server (also referred to as the 'Watch service') executes processes independently, after a Workflow configuration has been uploaded and the services have been started. The Workflow server can run only one configuration at a time.



There are a number of services related to Workflow. The Messenger service, for example, receives the files sent to Workflow from the Designer and the Workflow configuration tool.

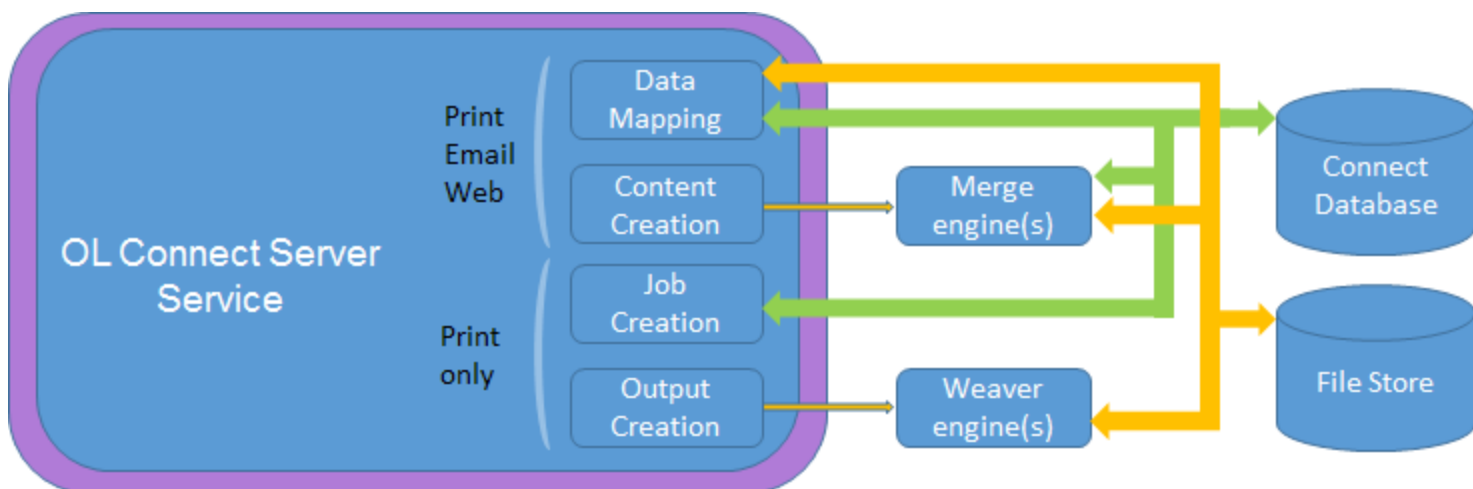
The Workflow Service Console lets you start and stop the different services, except the Connect server, and see their log files (see [Workflow Service Console](#)).

Note that Workflow isn't strictly limited to Connect functionality. It was originally developed as part of the PlanetPress Suite. Many of the plugins in the Workflow configuration tool are older than Connect; they were left in for compatibility reasons, even though they aren't all useful or usable within Connect. However, the Connect plugins cannot be used with the PlanetPress Suite software.

The Connect server

As opposed to the Workflow server, the Connect server was designed to be used only with Connect. The Connect server performs several different tasks, all of which are related to Connect content and content management:

- It communicates with the Workflow service (with the Connect plugins, specifically) and with the Designer when output is generated from the Designer.
- It creates records (by extracting data from a data source using a data mapping configuration), and jobs.
- It communicates with the engines (see below) in order to make them create content items and output (spool) files.



The Connect server is one of the components that has to be installed with Connect (see "Installation Wizard" on page 35).

In the Workflow Configuration Tool preferences you have to set the OL Connect server settings to enable Workflow to communicate with the server (see [Workflow Preferences](#)).

The **Connect Server Configuration** tool lets you change the settings for the Connect server, the engines and the service that cleans up the database and the file store. These settings can also be made in the preferences of the Designer.

The Connect database

The Connect database is the database back-end used by Connect itself when processing jobs. It can be either the MySQL instance provided by the Connect installer, or a pre-existing (external) instance (see "Database Considerations" on page 17).

All generated items (records, content items etc.) are stored in this database, for the next task in the process, as well as for future use, making it possible to commingle Print jobs, for example.

Note

Email content items are not stored in the Connect database.

A clean-up of the database is performed at regular intervals in accordance with the settings (see "Clean-up Service preferences" on page 697).

The File Store

Connect has its own File Store which it uses for transient files. The Clean-up service takes care of removing obsolete files when those files are not marked as permanent (see "Clean-up Service preferences" on page 697).

Tip

As of version 1.8, the File Store has become accessible for customer implementations. The Workflow configuration tool implements three tasks that allow you to Upload, Download and Delete files in the Connect File Store. The files can be accessed through the REST API, which means web portals could potentially access the files directly without having to go through a Workflow process.

The engines

Merge engine/s. A merge engine merges data with a template using the scripts in the template, in order to create (Print,Email or Web) content items.

The number of merge engines is configurable. By default, only one merge engine is used, but this number can be increased depending on the capacity of the machine that runs the solution (see "Performance Considerations" on page 25).

Weaver engines. The Weaver engines create Print output from Print content items. It takes the settings made in Print presets or in the Print Wizard into account. It also helps the data mapping engine by preparing any paginated input data.

The number of Weaver engines is configurable as well (see "Weaver Engine Scheduling" on page 88).

Speed units (parallels)

The number of 'speed units' is the maximum number of engines that can work in parallel, which is why they are also called 'parallels'. The output speed of all speed units together is limited to a certain number of output items (web pages, emails, or printed pages) per minute.

How many speed units you have and what the maximum total output speed will be is determined by your licence and any additional Performance Packs you might have.

There is one important twist: when generating **Print** output, the limit imposed by the number of speed units, only applies to the **Weaver** engines; when creating **Email** or **Web** output, the limit applies to the **Merge** engines only (the Weaver engine is not involved). Nevertheless, in situations where Print and Email or Web output are being created at the same time, all engines, regardless of their type, count towards the maximum number of speed units.

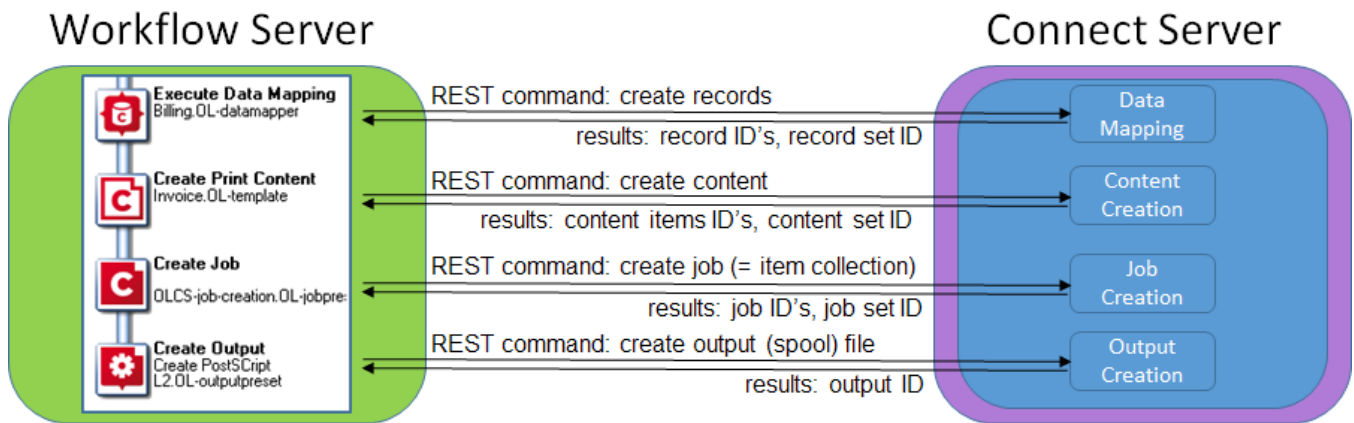
Each engine needs at least one speed unit. However, since the number of engines is configurable, and since small, medium and large jobs may run concurrently, the number of engines in use may not match the number of available speed units. When there are more speed units than there are engines in use, the Connect server distributes the speed units and the maximum output speed to the engines **proportionally**.

The REST API

The Connect server receives REST commands (see [The Connect REST API CookBook](#)), normally either via the Workflow service or from the Designer. This design allows the Connect functionality to be used by other applications. The server forwards the commands to the

appropriate engine and returns the results to the caller. The results are the id's of the items (records, content items, job etc.) that are stored in the Connect database (see below). All Connect tasks except the Create Web Content task integrate the results in the Metadata in Workflow.

The figure below shows the communication between Connect tasks and the Connect server in a Print process.



Printing and emailing from the Designer

To print or send email from within the Designer, the Connect service has to be running. The service is started automatically when the Designer starts, but it may not be running if the Connect Server and the Designer are installed on different computers. The Connect service can be found on the Services tab in the Task Manager.

For a proof print the Connect server is not used. Proof printing is always done locally, by the Designer.

Connect File Types

This article describes the different File Types that are related to PlanetPress Connect and its different modules. These are files that are generally transferable between machines, can be sent via email or other means.

- **.OL-template:** A Designer Template file, including up to 3 contexts. Is linked to a data mapping configuration by default, but not necessarily.

- **.OL-datamapper:** A Data Mapping Configuration file, which can include sample data (excluding database source files such as mySQL, oracle, etc).
- **.OL-datamodel:** A data model file which can be imported or exported into either a data mapping configuration or a template. Contains a list of fields and their data type (date, currency, string, etc).
- **.OL-jobpreset:** A job preset file, used when generating a job (ready for output) from Designer or through automation (Create Job task). Does sorting, splitting, adding metadata fields.
- **.OL-outputpreset:** An output preset file, used to generate the actual print output in the appropriate format (pcl, afp, pcl, ipds, pdf, etc). Includes print settings such as imposition (n-up, cut & stack), inserter marks, tray settings, etc.
- **.OL-package:** A transfer file used to package one or many of the above files (the data model being part of both the template and the data mapping configuration). Created by using the File -> Send to Workflow dialog, and choosing "File..." in the Destination box.
- **.OL-script:** One or more designer "scripts". Can be imported or exported from the Scripts pane in Designer when a template is open.
- **.OL-printerdef:** A Printer Definition File. Used by the Output Preset to determine what type of output to produce. These are generated by an internal application that is not currently distributed outside of OL, but the definition files themselves can be provided.
- **.OL-workflow:** A Workflow file used by PlanetPress Workflow. Equivalent to .pp7 files (they are, in fact, essentially the same format), containing the processes and such used by Workflow.

The DataMapper Module

The DataMapper is the tool to create a data mapping configuration.

A data mapping configuration file contains the information necessary for data mapping: the settings to read the source file (Delimiter and Boundary settings), the data mapping workflow with its extraction instructions ('Steps'), the Data Model and any imported data samples.

Data mapping configurations are used to extract data and transpose that data into a format that can be shared amongst different layouts and outputs created with the Connect Designer and Workflow.

The original data, located in a file or database outside of Connect, is called a data source.

The first step in the data extraction process is making settings for the input data (see "Data source settings" on page 115) , including boundaries for each record inside the data sample.

When you define the boundaries, you are actually defining a series of records inside your data sample file.

After configuring these settings you can start working on the logic to extract data from each of those records. You need to identify and extract data from each record. To achieve this, you will create a data mapping workflow, consisting of multiple steps (extractions, loops, conditions and more) (see "Data mapping workflow" on page 113 and "Extracting data" on page 118).

When this process is complete, the result is a Data Model. This model contains the necessary information to add variable data to Connect Designer templates. (see "The Data Model" on page 151 for more information). It has a generic format with an emphasis on content, free from any restrictions imposed by the file types or the origin of the data. This allows a same layout or output to be populated with data from different sources and formats without the need to modify it.

DataMapper basics

Connect's DataMapper lets you build a data mapping workflow to extract data from a variety of data sources. The data mapping workflow consists of multiple 'steps' which process and extract data from each record of a data source and store it in a new, extracted record set. The data mapping workflow is saved in a data mapping configuration.

- 1. Create a new data mapping configuration.**

Run the Designer and start creating a data mapping configuration by selecting a data source. See "Data mapping configurations" on the next page.

2. **Configure settings for the data source.**

The data source can be a file (CSV, PDF, TXT, XML) or a particular database. Configure how the data source is read by the DataMapper and create a record structure. See "Data source settings" on page 115.

3. **Build the data mapping workflow.**

A data mapping workflow always starts with the **Preprocessor** step and ends with the **Postprocessor** step. You can add as many steps as you like and edit the Data Model of the extracted data as required. See "Data mapping workflow" on page 113 and "The Data Model" on page 151.

What's next?

Use the data mapping configuration in the Designer module to create templates for personalized customer communications. To learn more, see "The Designer" on page 302.

In Workflow, a data mapping configuration can be used to extract data. The extracted data can then be merged with a Designer template to generate output in the form of print, email or a web page. See [Workflow](#) and [Connect tasks in Workflow](#).

Data mapping configurations

A data mapping configuration file contains the information necessary for data mapping: the settings to read the source file (Delimiter and Boundary settings), the data mapping workflow with its extraction instructions ('Steps'), the Data Model and any imported data samples.

Data mapping configurations are used in the **Designer** to help add variable data fields and personalization scripts to a template. In fact, only a Data Model would suffice (see "Importing/exporting a Data Model" on page 153). The advantage of a data mapping configuration is that it contains the extracted records to merge with the template, which lets you preview a template with data instead of field names.

It is also possible to generate output of a data mapping configuration directly from the Designer (see "Generating output" on page 953).

Ultimately data mapping configurations are meant to be used by the **Execute Data Mapping** task in Connect Workflow processes, to extract data from a particular type of data file. Typically the extracted data is then merged with a template to generate output in the form of print, email and/or a web page. To make this happen, the data mapping configuration as well as the

template and any print presets have to be sent to Workflow; see "Sending files to Workflow" on page 309.

Note

AFP input requires the CDP library. The library licence allows PlanetPress Connect to run up to 4 instances of that library on a given machine at a given time.

Creating a new data mapping configuration

A new data mapping configuration can be made with or without a wizard. When you open a data file with a DataMapper wizard, the wizard automatically detects a number of settings. You can adjust these settings. Next, the wizard automatically extracts as many data fields (or metadata, in case of a PDF/VT or AFP file) as it can, in one extraction step. Without a wizard you have to make the settings yourself, and configure the extraction workflow manually.

Note

The DataMapper doesn't use the data source directly, rather it uses a copy of that data: a **data sample**. Although the data sample is a copy, it is updated each time the data mapping configuration is opened or whenever the data sample is selected. More samples can be added via the Settings pane; see "Data samples" on page 211.

From a file

To start creating a data mapping configuration without a wizard, first select the data file. There are two ways to do that: from the Welcome screen and from the File menu.

- From the Welcome screen
 1. Click **Create a New Configuration**.
 2. From the **From a file** pane and select a file type (Comma Separated Values or Excel (CSV/XLSX/XLS), MS-Access, PDF/VT, Text or XML).
 3. Click the **Browse** button and open the file you want to work with (for a database,

you may have to enter a password).

4. Click **Finish**.
- From the File menu
 1. Click the **File** menu and select **New**.
 2. Click the **Data mapping Configuration** drop-down and select **Files** and then the file type (Comma Separated Values or Excel (CSV/XLSX/XLS), MS-Access, PDF/VT, Text or XML).
 3. Click **Next**.
 4. Click the **Browse** button and open the file you want to work with.
 5. Click **Finish**.

Note

- Excel files saved in "Strict Open XML" format are not supported yet.
- PCL and PostScript (PS) files are automatically converted to PDF format. When used in a production environment (a Connect Workflow process) this may influence the processing speed, depending on the available processing power.

After opening the file, you have to make settings for the input data (see "Data source settings" on page 115). Then you can start building the data extraction workflow.


With a wizard

Data mapping wizards are available for PDF/VT, AFP, XML, CSV and database tabular files, because these files are structured in a way that can be used to automatically set record boundaries.

The wizard for PDF/VT and AFP files cannot extract data, only metadata. After opening such a file with the wizard, you can build the data extraction workflow.

The other wizards use the Extract All step to extract data, but they cannot create detail tables, so they are less suitable for files from which you want to extract transactional data.

There are two ways to open a data file with a wizard: from the Welcome screen or from the File menu.

- From the **Welcome** screen
 1. Open the PlanetPress Connect **Welcome** page by clicking the  icon at the top right or select the **Help** menu and then **Welcome**.
 2. Click **Create a New Configuration**.
 3. From the **Using a wizard** pane, select the appropriate file type.
- From the **File** menu
 1. In the menu, click **File > New**.
 2. Click the **Data mapping Wizards** drop-down and select the appropriate file type.

The steps to take with the wizard depend on the file type. See:

- "Using the wizard for CSV and Excel files" on page 106
- "Using the wizard for databases" on page 108
- "Using the wizard for PDF/VT and AFP files" on page 111
- "Using the wizard for XML files" on page 112


Generating a counter

Instead of creating a data mapping configuration for a certain type of data file, you may create a data mapping configuration that only contains a series of sequential numbers. This is a solution if, for instance, you need to create sequential tickets or anything that has an ID that changes on each record.

Note

You can't join this configuration to another data file. It is just a counter to be applied on a static template.

To generate a counter:

- From the **Welcome** screen
 1. Open the PlanetPress Connect **Welcome** page by clicking the  icon at the top right or select the **Help** menu and then **Welcome**.

2. Click **Create a New Configuration**.
 3. From the **Using a wizard** pane, select **Generate counters**.
- From the **File** menu
 1. In the menu, click **File > New**.
 2. Click the **Data mapping Wizards** drop-down and select **Generate counters**.

You can set the following parameters:

- **Starting Value:** The starting number for the counter. Defaults to 1.
- **Increment Value:** The value by which to increment the counter for each record. For example, an increment value of 3 and starting value of 1 would give the counter values of 1, 4, 7, 10, [...]
- **Number of records:** The total number of counter records to generate. This is not the end value but rather the total number of actual records to generate.
- **Padding character:** Which character to add if the counter's value is smaller than the width.
- **Width:** The number of digits the counter will have. If the width is larger than the current counter value, the padding character will be used on the left of the counter value, until the width is equal to the set value. For example for a counter value of "15", a width of "4" and padding character of "0", the value will become "0015".
- **Prefix:** String to add before the counter, for example, adding # to get #00001. The prefix length is not counted in the width.
- **Suffix:** String to add after the counter. The suffix length is not counted in the width.

Opening a data mapping configuration

To open an existing data mapping configuration, in the [Menus](#), select **File > Open**. Make sure that the file type is either DataMapper files or Connect files. Browse to the configuration file to open, select it and click **Open**.

Alternatively, click on **File > Open Recent** to select one of the recently opened configuration files.

Saving a data mapping configuration

A Data Mapping Configuration file has the extension .OL-datamapper. The file contains the settings, the extraction workflow ('Steps'), the Data Model and the imported Data Samples (excluding database source files such as MySQL, oracle, etc).

To save a data mapping configuration:

- In the [Menus](#), click on **File > Save**, or click on **Save As** to save a **copy** of a data mapping configuration under a different name.
- In the [Toolbars](#), click the **Save** button.

If the data mapping configuration has not been saved before, you have to browse to the location where the data mapping configuration should be saved and type a name, then click **Save**.


Using the wizard for CSV and Excel files

The DataMapper wizard for CSV and Excel files helps you create a data mapping configuration for such files. The wizard automatically detects delimiters and extracts all data in one extraction step.

The wizard interprets each line in the file as a record. If your data file contains transactional data, you will probably want more lines to go in one record and put the transactional data in detail tables.

The wizard cannot create detail tables. If the file contains transactional data, the data mapping configuration is best created without a wizard (see "Creating a new data mapping configuration" on page 102).

There are two ways to open a CSV file or Excel file with a wizard: from the Welcome screen or from the File menu.

- From the **Welcome** screen
 1. Open the PlanetPress Connect **Welcome** page by clicking the  icon at the top right, or select the **Help** menu and then **Welcome**.
 2. Click **Create a New Configuration**.
 3. From the **Using a wizard** pane, select **CSV/XLSX/XLS**.

4. Click the **Browse** button and open the file you want to work with.
 5. Click **Next**.
- From the **File** menu
 1. In the menu, click **File > New**.
 2. Click the **Data mapping Wizards** drop-down and select **From CSV/XLSX/XLS File**.
 3. Click **Next**.
 4. Click the **Browse** button and open the file you want to work with.
 5. Click **Next**.

After selecting the file, take a look at the **preview** to ensure that the file is the right one and the encoding correctly reads the data. Click **Next**.

For an **Excel** file you can indicate whether or not the first row contains field names, and from which sheet the data should be extracted.

Note

Excel files saved in "Strict Open XML" format are not supported yet.

For a CSV file the will display the different settings it has detected, allowing you to change them:

- **Encoding**: Defines which encoding is used to read the file.
- **Separator**: Defines which character separates each field in the file.
- **Comment Delimiter**: Defines which character starts a comment line.
- **Text Delimiter**: Defines which character surrounds text fields in the file. Separators and comment delimiters within text are not interpreted as separator or delimiter; they are seen as text.
- **Ignore unparseable lines**: Ignores any line that does not correspond to the settings above.

- **First row contains field names:** Uses the first line of the CSV as headers, which automatically names all extracted fields.

Verify that the data are read properly.

Finally click **Finish**. All data fields are automatically extracted in one extraction step.


Using the wizard for databases

The DataMapper wizard for database files helps you create a data mapping configuration for a database file. The wizard extracts the data in one extraction step.

The wizard cannot create detail tables. If the file contains transactional data, the data mapping configuration is best created without a wizard (see "Creating a new data mapping configuration" on page 102).

Opening a database file with a wizard

There are two ways to open an XML file with a wizard: from the Welcome screen or from the File menu.

- From the **Welcome** screen
 1. Open the PlanetPress Connect **Welcome** page by clicking the  icon at the top right or select the **Help** menu and then **Welcome**.
 2. Click **Create a New Configuration**.
 3. From the **Using a wizard** pane, select **Database**.
 4. Use the drop-down to select the database type.
 5. Click **Next**.
- From the **File** menu
 1. In the menu, click **File > New**.
 2. Click the **Data mapping Wizards** drop-down and select **From databases**.
 3. Click **Next**.
 4. Use the drop-down to select the database type.
 5. Click **Next**.

Wizard settings for a database file

After opening a database file with a wizard there are a number of settings to make, depending on the database type (see below).

On the last page of the dialog, click **Finish** to close the dialog and open the actual data mapping configuration.

MySQL, SQL Server or Oracle

- **Server:** Enter the server address for the database.
- **Port:** Enter the port to communicate with the server. The default port is 3306.
- **Database name:** Enter the exact name of the database from where the data should be extracted.
- **User name:** Enter a user name that has access to the server and specified database. The user only requires **Read** access to the database.
- **Password:** Enter the password that matches the user name above.
- **Table name:** The selected database is a set of related tables composed of rows and columns corresponding respectively to source records and fields. Select a table from which you want to extract data.
- **Encoding:** Choose the correct encoding to read the file.

Microsoft Access

- **Password:** Enter a password if one is required.
- **Table name:** The selected database is a set of related tables composed of rows and columns corresponding respectively to source records and fields. Select a table from which you want to extract data.
- **Encoding:** Choose the correct encoding to read the file.

ODBC Data Source

- **ODBC Source:** Use the drop-down to select an ODBC System Data Source. This must be a data source that has been configured in the 64-bit ODBC Data Source Administrator, as PlanetPress Connect is a 64-bit application and thus cannot access 32-bit data sources.

- **This ODBC source is MSSQL:** Check this option if the ODBC source is MSSQL (SQL Server). The options below appear under **MSSQL-ODBC advanced configuration**:
 - **Windows authentication:** Select to use the Windows user name and password that are used by the Connect Service.
 - **SQL Server authentication:** Select to use the User name and Password set below to connect to the SQL Server:
 - **User name:** Enter the SQL Server user name.
 - **Password:** Enter the password for the above user name.

JDBC

Note

Since JDBC can connect to multiple types of databases, a specific database driver and path to this driver's JAR file must be specified.

- **JDBC Driver:** Use the drop-down to select which JDBC Driver to use for the database connection.
- **JAR file path:** Enter a path to the JAR file that contains the appropriate driver for the database.
- **Server:** Enter the server address for the database server.
- **Database name:** Enter the exact name of the database from where the data should be extracted.
- **User name:** Enter a user name that has access to the server and specified database. The user only requires **Read** access to the database.
- **Password:** Enter the password that matches the user name above.
- **Advanced mode:** Check to enable the **Connection String** field to manually enter the database connection string.
- **Connection string:** Type or copy in your connection string.

Using the wizard for PDF/VT and AFP files

The pages in PDF/VT and AFP files can be grouped on several levels. Additional information can be attached to each level in the structure. The structure and additional information are stored in the file's metadata.


The DataMapper wizard for PDF files lets you select a level to trigger the start of a new record and it also enables you to extract the additional information from the metadata. You can extract data from the content afterwards.

Tip

To extract information from the metadata in the extraction workflow itself, you have to create a JavaScript extraction (see "Using scripts in the DataMapper" on page 255 and "extractMeta()" on page 277).

If the PDF doesn't contain any metadata, each page is a new record - in other words, a boundary is set at the start of a new page -, which is exactly what happens when you open the file without a wizard.

You can open a PDF/VT file with a wizard using the Welcome screen or the File menu.

- From the **Welcome** screen
 1. Open the PlanetPress Connect **Welcome** page by clicking the  icon at the top right or select the **Help** menu and then **Welcome**.
 2. Click **Create a New Configuration**.
 3. From the **Using a wizard** pane, select **PDF/VT**.
 4. Click the **Browse** button and open the PDF/VT file you want to work with. Click **Next**.
- From the **File** menu
 1. In the menu, click **File > New**.
 2. Click the **Data mapping Wizards** drop-down and select **From PDF/VT or AFP**.
 3. Click **Next**.
 4. Click the **Browse** button and open the PDF/VT file you want to work with. Click **Next**.

After selecting the file, select the following options in the **Metadata** page:

- **Metadata record levels:** Use the drop-down to select what level in the metadata defines a record.
- **Field List:** This list displays all fields on the chosen level and higher levels in the PDF/VT metadata. The right column shows the field name. The left column displays the level on which it is located. Check any field to add it to the extraction.

Click **Finish** to close the dialog and open the actual Data Mapping configuration.


On the Settings pane, you will see that the boundary trigger is set to **On metadata**. The selected metadata fields are added to the Data Model.

Using the wizard for XML files

The DataMapper wizard for XML files helps you create a data mapping configuration for an XML file. The wizard lets you select the type of node and the trigger that delimit the start of a new record. Next, the wizard extracts the data in one extraction step.

The wizard cannot create detail tables. If the file contains transactional data, the data mapping configuration is best created without a wizard (see "Creating a new data mapping configuration" on page 102).

There are two ways to open an XML file with a wizard: from the Welcome screen or from the File menu.

- From the Welcome screen
 1. Open the PlanetPress Connect **Welcome** page by clicking the  icon at the top right or select the **Help** menu and then **Welcome**.
 2. Click **Create a New Configuration**.
 3. From the **Using a wizard** pane, select XML.
 4. Click the **Browse** button and open the XML file you want to work with. Click **Next**.
- From the File menu
 1. In the menu, click **File > New**.
 2. Click the **Data mapping Wizards** drop-down and select **From XML File**.

3. Click **Next**.
4. Click the **Browse** button and open the XML file you want to work with. Click **Next**.

After selecting the file, you have to set the split level and trigger type:

- **XML Elements:** This is a list of node elements that have children nodes. Select the level in the data that will define the source record.
- **Trigger:** select **On element** to create a record in the Data mapping for each occurrence of the node element selected in the **XML Elements** field, or select **On change** to create a record each time the element is different. (Check the option to include attributes in the list of content items that can trigger a boundary.)

Note

The DataMapper only extracts elements for which at least one value is defined in the file. Attribute values are not taken into account.

Attribute values (prefixed with an @ sign in the Data Viewer) are not extracted automatically.

Click **Finish** to close the dialog and open the data mapping configuration.

Data mapping workflow

A **data mapping workflow** is a series of extraction instructions, called **steps**. These steps process and extract the data from the source and store them in records, of which the structure is determined in the Data Model (see "The Data Model" on page 151). Together with the data source settings, the Data Model, and the sample data, this is what makes a data mapping configuration (See "Data mapping configurations" on page 101).

The data mapping workflow is shown on the Steps pane at the left (see "Steps pane" on page 213).

Creating a data mapping workflow

A data mapping workflow always starts with the **Preprocessor** step and ends with the **Postprocessor** step. These steps allow the application to perform actions on the data file itself before it is handed over to the data mapping workflow ("Preprocessor step" on page 140) and

after the Data Mapping workflow has completed ("Postprocessor step" on page 150). When you create a new data mapping configuration, these steps are added automatically, but they don't actually do anything until you configure these steps. In between the Preprocessor and Postprocessor step, the workflow can contain as many steps as needed to extract the required data.

Adding steps

Extracting data is the main way to build a data mapping workflow; see "Extracting data" on page 118.

Extract steps, Condition steps and Repeat steps can be added after selecting data in the Data Viewer.

All steps can be added via the Steps pane:

1. In the extraction workflow on the **Steps** pane, select the step after which to add the new step.
2. Right-click on the Steps pane and select **Add a Step**; then select one of the step types.

Editing steps



The properties of each step in the extraction workflow become visible in the **Step properties** pane when you select that step in the Steps pane.

The **name** of each step is shown in the Steps pane. You can change it under **Description** in the Step properties pane.

The other properties are different per step type; see "Steps" on page 140.

Rearranging steps

To rearrange steps, simply drag & drop them somewhere else on the dotted line in the Steps pane.

Alternatively you can right-click on a step and select **Cut Step** or use the Cut  button in the **Toolbar**. If the step is **Repeat** or **Condition**, all steps under it will also be placed in the clipboard. To place the step at its destination, right-click the step in the position before the desired location and click **Paste Step**, or use the Paste  button in the toolbar.

Keep in mind that steps may influence each other, so you may have to move other steps as well to ensure that the workflow continues to function properly. In a Text file for example, an Extract step may need a Goto step before it that moves the cursor to a certain place in the source data.

Deleting steps

To delete a step, right-click on it in the Steps pane and select **Delete Step**.

Testing the extraction workflow

The extraction workflow is always performed on the current record in the data source. When an error is encountered, the extraction workflow stops, and the field on which the error occurred and all subsequent fields will be greyed out. Click the **Messages** tab (next to the Step properties pane) to see any error messages.

To test the extraction workflow on **all** records, you can:

- Click the **Validate All Records** toolbar button.
- Select **Data > Validate Records** in the menu.

If any errors are encountered in one or more records, an error message will be displayed. Errors encountered while performing the extraction workflow on the current record will also be visible on the **Messages** tab.

Data source settings

After opening a data file you have to make a number of settings to make sure that the source data is interpreted and grouped the way you want. These settings are found on the **Settings** pane at the left.

- **Input Data settings** help the DataMapper read the data source and recognize data correctly.
- **Boundaries** mark the start of a new record. They let you organize the data, depending on how you want to use them.
- **Data format settings** define how dates, times and numbers are formatted in the data source.

Input data settings (Delimiters)

The **Input Data** settings (on the **Settings** pane at the left) specify how the input data must be interpreted. These settings are different for each data type. For a CSV file, for example, it is important to specify the delimiter that separates data fields. PDF files are already delimited

naturally by pages, so the input data settings for PDF files are interpretation settings for text in the file.

For an overview of all options, see: "Input Data" on page 203.

For a CSV File

In a CSV file, data is read line by line, where each line can contain multiple fields, separated by a **delimiter**. Even though CSV stands for comma-separated values, fields may be separated using any character, including commas, tabs, semicolons, and pipes.

The **text delimiter** is used to wrap around each field just in case the field values contain the field separator. This ensures that, for example, the field "Smith; John" is not interpreted as two fields, even if the field delimiter is the semicolon.

For an explanation of all the options, see: "CSV file Input Data settings" on page 203.

For a PDF File

PDF files have a clear and unmovable delimiter: pages. So, the Input Data settings are not used to set delimiters. Instead, these options determine how words, lines and paragraphs are detected when you select content in the PDF to extract data from it.

For an explanation of all the options, see: "PDF file Input Data settings" on page 204.

For a database

Databases all return the same type of information. Therefore the Input Data options for a database refer to the tables inside the database. Clicking on any of the tables shows the first line of the data in that table.

If the database supports stored procedures, including inner joins, grouping and sorting, you can use custom SQL to make a selection from the database, using whatever language the database supports.

For an explanation of all the options, see: "Database Input Data settings" on page 204.

For a text file

Because text files have many different shapes and sizes, there are a lot of input data settings for these files. You can add or remove characters in lines if it has a header you want to get rid of, or strange characters at the beginning of your file, for example; you can set a line width if you are still working with old line printer data; etc.

It is important that pages are defined properly. This can be done either by using a set number of lines or using a text (for example, the character "P"), to detect on the page. Be aware that this is not a Boundary setting; it detects each new page, not each new record.

For an explanation of all the options, see: "Text file Input Data settings" on page 205.

For an XML file

XML is a special file format because these file types can have a theoretically unlimited number of structure types. The input data has two simple options that basically determine at which node level a new record is created. You can either select an element type, to create a new delimiter every time that element is encountered, or choose to use the root node. If there is only one top-level element, there will only be one record before the Boundaries are set.

Note

The DataMapper only extracts elements for which at least one value is defined in the file.

See also: "XML File Input Data settings" on page 206.

Record boundaries

Boundaries are the division between **records**: they define where one record ends and the next record begins. Using boundaries, you can organize the data the way you want.

You could use the exact same data source with different boundaries in order to extract different information. If, for instance, a PDF file contains multiple invoices, each invoice could be a record, or all invoices for one customer could go into a single record.

Keep in mind that when the data is merged with a template, each **record** generates output (print, email, web page) for a **single** recipient.

To set a boundary, a specific **trigger** must be defined.

The trigger can be a natural delimiter between blocks of data, such as a row in a CSV file or a page in a PDF file.

It can also be something in the data that is either static (for example, the text "Page 1 of" in a PDF file) or changing (a customer ID, a user name, etc).

To define a more complex trigger you could write a script (see "Setting boundaries using JavaScript" on page 257).

A new record cannot start in the middle of a data field, so if the trigger is something in the data, the boundary will be set on the nearest preceding natural delimiter. If for instance in a PDF file the text "Page 1 of" is used as the trigger, the new record starts at the page break before that text.

For an explanation of all Boundaries options per file type, see "Boundaries" on page 207.

Data format settings

By default the data type of extracted data is a String, but each field in the Data Model can be set to contain another data type (see "Data types" on page 168). When that data type is Date, Number or Currency, the DataMapper will expect the data in the data source to be formatted in a certain way, depending on the settings.

The default format for dates, numbers and currencies can be set in three places: in the user preferences, in the data source settings, and per field in the Data Model.

By default, the user preferences are set to the system preferences. These user preferences become the default format values for any newly created data mapping configuration. To change these preferences, select **Window > Preferences > DataMapper > DataMapper default format** (see "Datamapper preferences" on page 703).

Data format settings defined for a data source apply to any **new** extraction made in the **current** data mapping configuration. These settings are made on the Settings pane; see "Settings pane" on page 203.

Settings for a field that contains extracted data are made via the properties of the Extract step that the field belongs to (see "Setting the data type" on page 159). Any format settings specified per field are always used, regardless of the user preferences or data source settings.

Note

Data format settings tell the DataMapper how certain types of data are formatted **in the data source**. They don't determine how these data are formatted in the Data Model or in a template. In the Data Model, data are converted to the native data type. Dates, for example, are converted to a DateTime object in the Data Model, and will always be shown as "year-month-day" plus the time stamp, for example: 2012-04-11 12.00 AM.

Extracting data

Data are extracted via Extraction steps into fields in the Data Model. This topic explains how to do that. Fields can also be filled with other data: the result of a JavaScript or the value of a property. To learn how to do that, see "Fields" on page 155.

Before you start

Data source settings

Data source settings must be made beforehand, not only to make sure that the data is properly read but also to have it organized in a record structure that meets the purpose of the data mapping configuration (see "Data source settings" on page 115). It is important to set the **boundaries** before starting to extract data, especially transactional data (see "Extracting transactional data" on page 124). Boundaries determine which data blocks - lines, pages, nodes - form a record in the source data. Data that are located in different records cannot be put into the same record in the record set that is the result of the extraction workflow.

Preprocessor step

The Preprocessor step allows the application to perform actions on the data file itself before it is handed over to the Data Mapping workflow. In addition, properties can be defined in this step. These properties may be used throughout the extraction workflow. For more information, see "Preprocessor step" on page 140.

Adding an extraction

In an extraction workflow, Extract steps are the pieces that take care of the actual data extractions.

To add an Extract step:


1. In the Data Viewer pane, select the data that needs to be extracted. (See "Selecting data" on page 122.)
2. Choose one of two ways to extract the selected data.
 - Right-click on the selected data and select **Add Extraction** from the contextual menu.

Note

For optimization purposes, it is better to add data to an existing Extract step than to have a succession of extraction steps. To do that, select that step on the Steps pane first; then right-click on the selected data and choose **Add Extract Field**.

- Alternatively, **drag & drop** the selected fields into the Data Model pane.

Tip

In a PDF or Text file, use the Drag icon  to drag selected data into the Data Model.

With this method, a new Extract step will only be added to the extraction workflow when no Extract step already present on the Steps pane. Otherwise the field/s will be added to the selected Extract step or to the one that was last added.

Dragging data into an **existing field** in the Data Model will replace the data. The field name stays the same.

Drop data on empty fields or on the **record** itself to add new fields.

Special conditions

The Extract step may need to be combined with another type of step to get the desired result.

- Data can be extracted **conditionally** with a Condition step or Multiple Conditions step; see "Condition step" on page 145 or "Multiple Conditions step" on page 148.
- Normally the same extraction workflow is automatically applied to all records in the source data. It is however possible to **skip records** entirely or partially, using an Action step. Add an Action step in a branch under a Condition step or Multiple Conditions step (see "Action step" on page 149) and set the type of action to Stop Processing Record (see "Text and PDF Files" on page 226).
- To extract **transactional** data, the Extract step must be placed inside a Repeat step. See "Extracting transactional data" on page 124.

Note

Fields cannot be used twice in one extraction workflow.

Different Extract steps can only write extracted data to the **same** field in the Data Model, if:

- The field name is the same. (See: "Renaming and ordering fields" on page 158.)
- The Extract steps are mutually exclusive. This is the case when they are located in different branches of a Condition step or Multiple Conditions step.
- The option **Append values to current record** is checked in the Step properties pane under Extraction Definition.

Extracting data into multiple fields

When you select multiple fields in a CSV or tabular data file and extract them simultaneously, they are put into different fields in the Data Model automatically.

In a PDF or Text file, when multiple lines are extracted at the same time, they are by default joined and put into one field in the Data Model. To split them and put the data into different fields:

1. Select the field in the Data Model that contains the extracted lines.
2. On the **Step properties** pane, under **Field Definition**, click the drop-down next to **Split** and select **Split lines**.

Adding fields to an existing Extract step

For optimization purposes, it is better to add fields to an existing Extract step than to have a succession of extraction steps.

To add fields to an existing Extract step:

1. In the Data Viewer pane, select the data that needs to be extracted. (See "Selecting data" on the facing page.)
2. Select an **Extract** step on the **Steps** pane.
3. Right-click on the data and select **Add Extract Field**, or drag & drop the data on the Data Model.

When data are dropped on the Data Model, they are by default added to the last added Extract step.

Editing fields

After extracting some data, you may want to:

- Change the **names** of fields that are included in the extraction.
- Change the **order** in which fields are extracted.
- Set the **data type**, **data format** and **default value** of each field.
- Modify the extracted data through a **script**.
- Delete a field.

All this can be done via the Step properties pane (see "Settings for location-based fields in a Text file" on page 221), because the fields in the Data Model are seen as properties of an Extract step. See also: "Fields" on page 155.

Testing the extraction workflow

The extraction workflow is always performed on the current record in the data source. When an error is encountered, the extraction workflow stops, and the field on which the error occurred and all subsequent fields will be greyed out. Click the **Messages** tab (next to the Step properties pane) to see any error messages.

To test the extraction workflow on **all** records, you can:

- Click the **Validate All Records** toolbar button.
- Select **Data > Validate Records** in the menu.

If any errors are encountered in one or more records, an error message will be displayed. Errors encountered while performing the extraction workflow on the current record will also be visible on the **Messages** tab.

Selecting data

In order to extract the data, it is necessary to first define the data to be extracted, by selecting it. It depends on the data source file type how this is done. The following paragraphs explain how to create and manipulate a data selection in each different type of file.

Data selections are used to extract promotional data ("Extracting data" on page 118), transactional data ("Extracting transactional data" on page 124) and to apply a condition to an extraction (Condition step).

Right-clicking on a data selection displays a contextual menu with the actions that can be done with that selection or the steps that can be added to them. That menu also displays the keyboard shortcuts.

Text or PDF file

To **select** data in a Text or PDF file, click on a starting point, keep the mouse button down, drag to the end of the data that needs to be selected and release the mouse button. The data selection can contain multiple lines.

To **resize** a data selection, click and hold on one of the resize handles on the borders or corners, move them to the new size and release the mouse button.

To **move** the data selection, click and hold anywhere on the data selection, move it to its new desired location and release the mouse button.

Note

In a Text or PDF file, when you move the selection rectangle directly after extracting data, you can use it to select data for the next extraction.

However, moving the selection rectangle that appears after clicking on a field in the Data Model actually changes which data is extracted into that field.

CSV file or database data

Tabular data is displayed in the Data Viewer in a table where multiple fields appear for each line or row in the original data.

To select data, click on a field, keep the mouse button down, drag to the last field that you want to select and release the mouse button.

Alternatively you can select fields just like files in the Windows Explorer: keep the Ctrl button pressed down while clicking on fields to select or deselect them, or keep the Shift button pressed down to select consecutive fields.

XML File

XML data is displayed as a tree view inside the Data Viewer. To get a better overview you can also collapse any XML level.

In this tree view you can select nodes just like files in the Windows Explorer: keep the Ctrl

button pressed down while clicking on nodes to select or deselect them, or keep the Shift button pressed down to select consecutive nodes.

You can select multiple fields even if those fields are in different nodes.

Note

The Goto step isn't used in XML extraction workflows. The DataMapper moves through the file using **Xpath**, a path-like syntax to identify and navigate nodes in an XML document.

Extracting transactional data

Promotional data are data about customers, such as addresses, names and phone numbers. In Connect, each record in the extracted record set represents one recipient. The number of fields that contain promotional data is the same in each record. These data are stored on the root level of the extracted record.

Transactional data, on the other hand, are used in communications about transactions between a company and their customers or suppliers: invoices, statements, and purchase orders, for example. Naturally these data differ per customer. They are stored in **detail tables** in the extracted record. The number of fields in a detail table can vary from record to record.

abc	SubTotal	93.76
abc	TaxTotal	14.04
abc	Total	107.80
▲	services [1]	1
abc	Number	TVDIGHD
abc	Description	High Definition Digital TV
abc	BasePrice	30.00
abc	Rebates	-28.19
abc	Charges	67.25
abc	SubTotal	39.06
abc	AccountNumber	6784589574527852
abc	UserName	
abc	Type	Television
▲	charges [8]	1
abc	ID	TVDIGBS
abc	Description	TB Basic Digital Service
abc	Price	30.00
▲	details [6]	1
abc	CODE	MVHDLOC
abc	ID	55123
abc	Description	Gangster Squad HD
abc	TimeStamp	2013-02-16 19:24:33
abc	Price	5.95

Detail tables are created when an **Extract** step is added within a **Repeat** step. The Repeat step goes through a number of lines or nodes. An Extract step within that loop extracts data from each line or node.

It depends on the type of source data how this loop is constructed exactly.

For more information about detail tables, **multiple** detail tables and **nested** detail tables, see "Example " on page 196.

From a CSV file or a Database

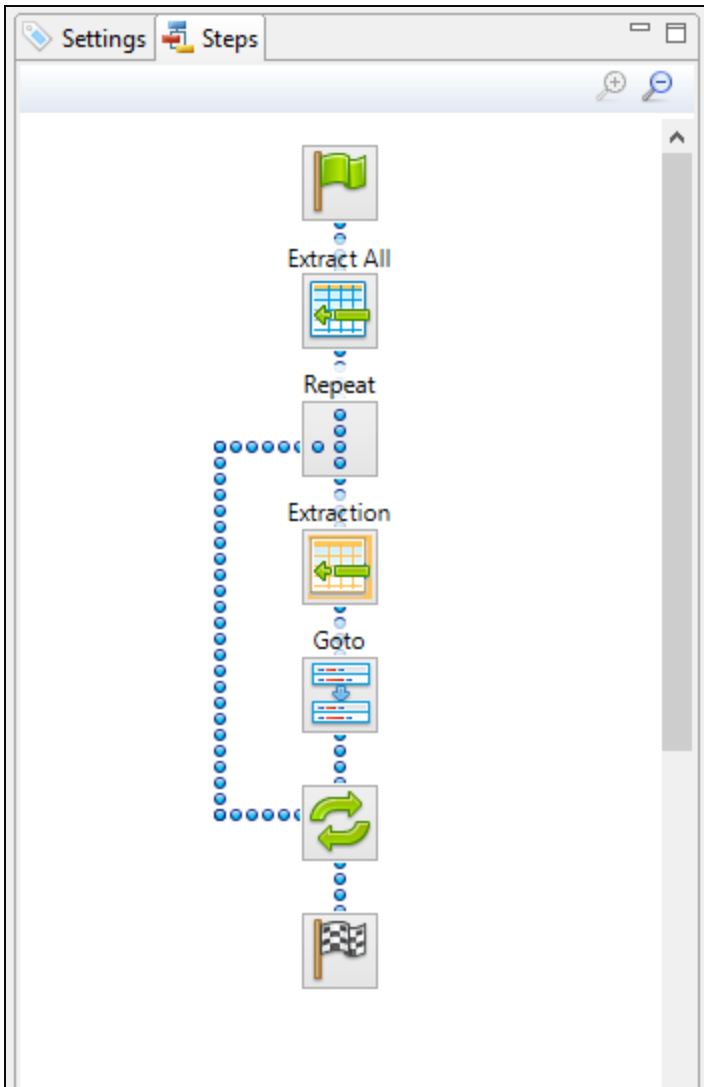
The transactional data (also called line items) appear in multiple rows.

1. Select a field in the column that contains the first line item information.
2. Right-click this data selection and select **Add Repeat**.

ItemNumber	ItemDesc	ItemUnitPrice	ItemOrdered	ItemShipped	ItemBO	ItemTotal	TotalOrdered	TotalShipped	TotalBO
PSM005	Upgrade (Starter to Web)	7495	3	3	0	22485	26	26	0
PSM010	Document Design Training - 1 day	1250	3	3	0	3750	26	26	0
PSM003	PSM Web	7995	2	2	0	15990	26	26	0
PSM002	PSM Production (unlimited)	4995	1	1	0	4995	26	26	0
PSM004	Upgrade (Starter to Production)	4495	3	3	0	13485	26	26	0
PSM008	B2C and Payment Module Upgrade	795	2	2	0	1590	26	26	0
PSM011	PSM (price per class) - 1 day	2000	3	3	0	6000	26	26	0
PSM006	Upgrade (Production to Web)	4375	2	2	0	8750	26	26	0
PSM001	PSM Starter (100K records)	995	3	3	0	2985	26	26	0
PSM007	Additional 350K records	2500	3	3	0	7500	26	26	0
PSM009	e-Learning - PSM Training program	595	1	1	0	595	26	26	0

This adds a Repeat step with a GoTo step inside it. The GoTo step moves the cursor down to the next line, until there are no more lines (see "Goto step" on page 144).

3. (Optional.) Add an empty detail table via the Data Model pane: right-click the Data Model and select **Add a table**. Give the detail table a name.
4. Select the Repeat step on the Steps pane.
5. Start extracting data (see "Adding an extraction" on page 119).
When you drag & drop data on the name of a detail table in the Data Model pane, the data are added to that detail table.
Dropping the data somewhere else on the Data Model pane creates a new detail table, with a default name that you can change later on (see "Renaming a detail table" on page 193).
The extraction step is placed inside the Repeat step, just before the GoTo step.



The screenshot shows a data mapping tool interface with three main panels:

- Tabular viewer:** Displays a table of data with columns: guage, Brand, PW, ItemNumber, ItemDesc, ItemUnitPrice, ItemOrdered, ItemShipped, ItemBO, ItemTotal, TotalOrdered, and TotalShip. The data includes various items like 'Upgrade (Starter to Web)', 'Document Design Training', and 'PSM Production (unlimited)'. A blue callout 'Transactional data' points to the 'ItemOrdered' column.
- Detail table:** Located on the right, it shows a detailed view of a single record (record [111]). A red circle highlights a section of the detail table with columns: ItemNumber, ItemDesc, ItemUnitPrice, ItemOrdered, ItemShipped, ItemBO, and ItemTotal. A blue callout 'Detail table' points to this section.
- Step Properties:** Located at the bottom, it shows the configuration for a 'Repeat step'. The 'Repeat definition' is set to 'Until no more elements'. A blue callout 'Condition to end the loop' points to this setting.

On the left side, a 'Steps' pane shows a workflow with 'Goto and Extract steps' highlighted by a blue callout.

From an XML file

The transactional data appears in repeated elements.

1. Right-click one of the repeating elements and select **Add Repeat**.

The screenshot shows the data mapping tool interface with two main panels:

- XML viewer:** Displays an XML tree structure. The root element is 'Rep (REP0290771)'. It contains several sub-elements: 'SubTotal (45184.98)', 'TaxTotal (3388.87)', 'Total (48573.85)', and three 'ITEM ()' elements. Each 'ITEM' element contains details like 'Number', 'Description', 'UnitPrice', 'Ordered', 'Shipped', 'BackOrder', and 'Total'. A blue callout 'Add Repeat' points to one of the 'ITEM' elements.
- Data model:** Located on the right, it shows a table of data with columns: Name and Value. The data includes fields like 'FULLNAME', 'ID', 'Gender', 'LastName', 'FirstName', 'Address1', 'Address2', 'City', 'State', 'Country', 'ZipCode', 'Title', 'Company', 'Phone2', 'Email', 'Language', 'MDS', 'LINE_ITEMS', 'Number', 'Date', 'DueDate', 'Rep', 'SubTotal', 'TaxTotal', and 'Total'. A blue callout 'Add Repeat' points to the 'LINE_ITEMS' field.

This adds a **Repeat step** to the data mapping configuration.

By default, the **Repeat type** of this step is set to **For Each**, so that each of the repeated

elements is extracted. You can see this on the **Step properties** pane, as long as the Repeat step is selected on the Steps pane. In the **Collection** field, you will find the corresponding node path.

Tip

It is possible to edit the Xpath in the Collection field, to include or exclude elements from the loop. One example of this is given in a How-to: [Using Xpath in a Repeat step](#).

The example in the How-to uses the `starts-with()` function. For an overview of XPath functions, see [Mozilla: XPath Functions](#).

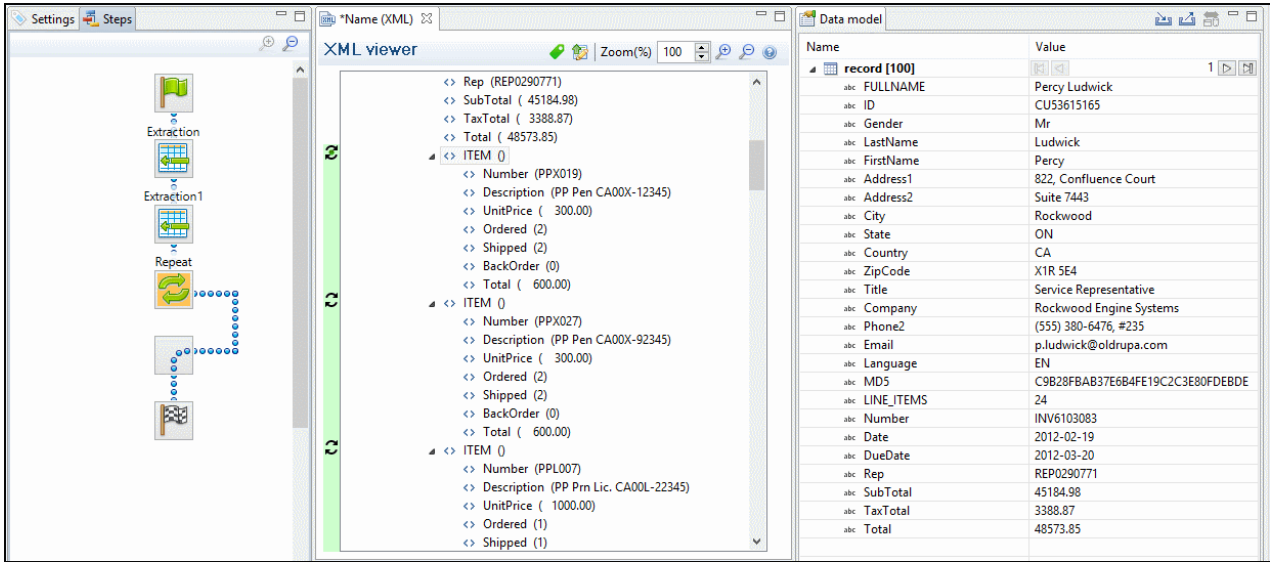
The Goto step isn't used in XML extraction workflows The DataMapper moves through the file using **Xpath**, a path-like syntax to identify and navigate nodes in an XML document.

2. (Optional.) Add an empty detail table via the Data Model pane: right-click the Data Model and select Add a table. Give the detail table a name.
3. Select the Repeat step on the Steps pane.
4. Extract the data: inside a repeating element, select the data that you want to extract. Then right-click the selected nodes and select **Add Extraction**, or drag & drop them in the Data Model.

When you drag & drop data on the name of a detail table in the Data Model pane, the data are added to that detail table.

Dropping the data somewhere else on the Data Model pane creates a new detail table, with a default name that you can change later on (see "Renaming a detail table" on page 193).

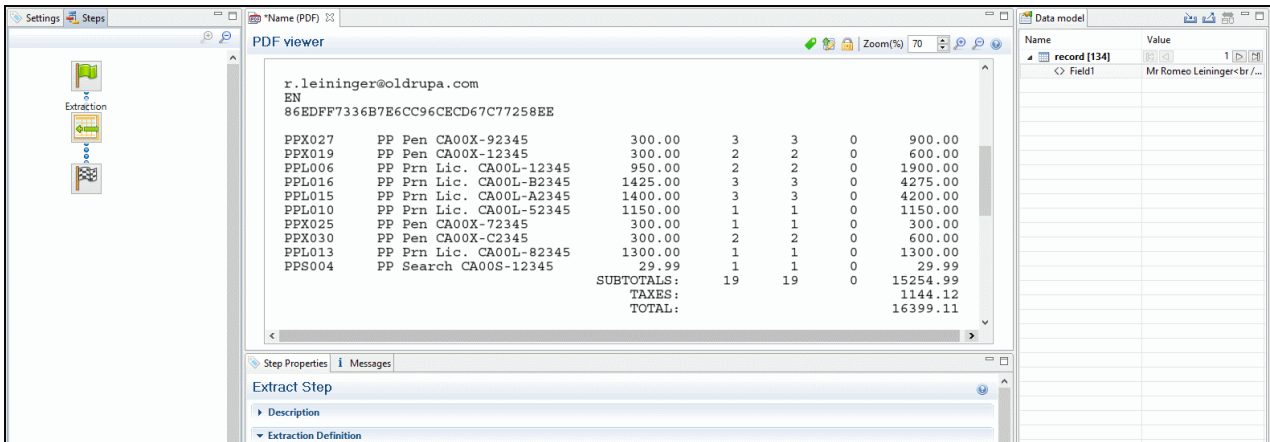
The new **Extract** step will be located in the Repeat step.



From a Text or a PDF file

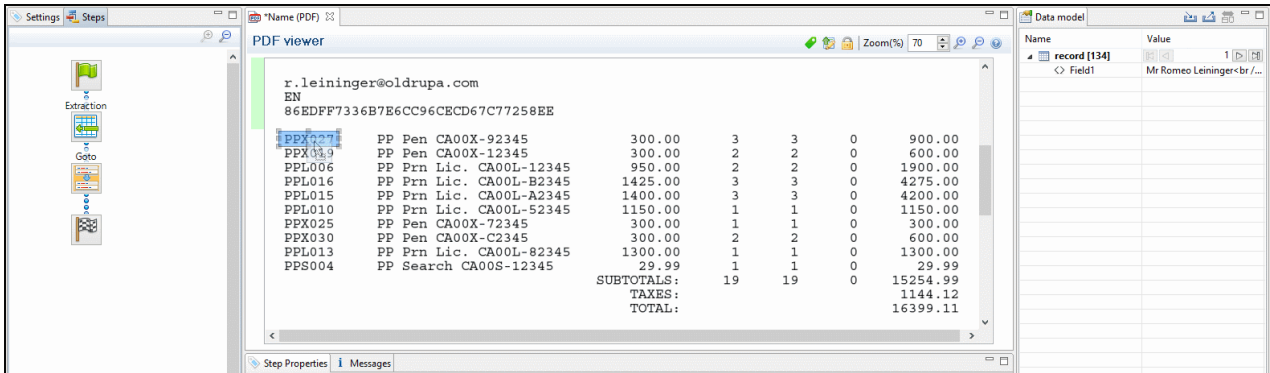
In a PDF or Text file, transactional data appears on multiple lines and can be spread over multiple pages.

1. **Add a Goto step if necessary.** Make sure that the cursor is located where the extraction loop must start. By default the cursor is located at the top of the page, but previous steps may have moved it. Note that an Extract step does not move the cursor.
 1. Select something in the first line item.
 2. Right-click on the selection and select **Add Goto**. The Goto step will move the cursor to the start of the first line item.



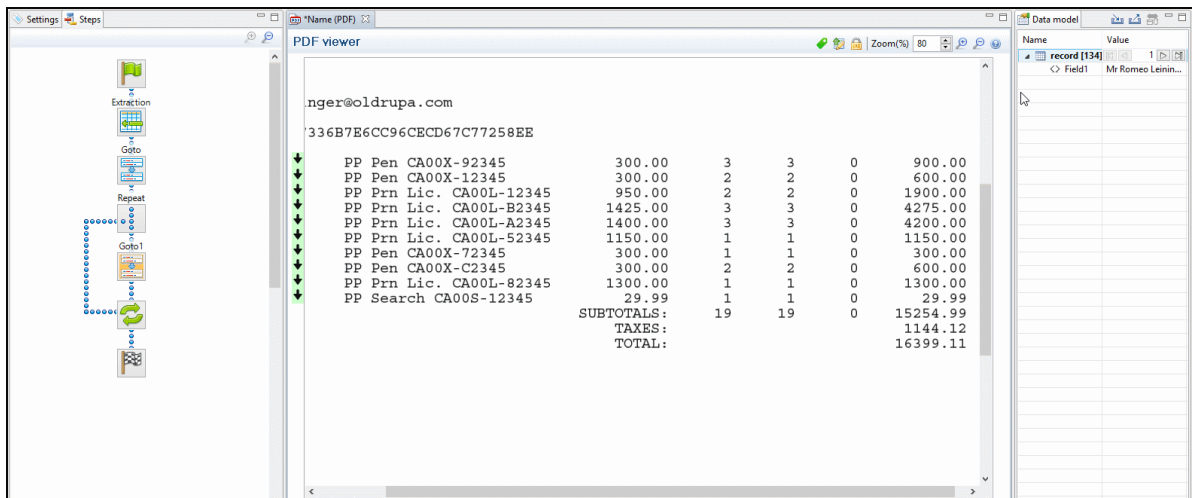
2. Add a **Repeat** step where the loop must stop.

1. In the line under the last line item, look for a text that can be used as a condition to stop the loop, for example "Subtotals", "Total" or "Amount".
2. Select that text, right-click on it and select **Add Repeat**. The Repeat step loops over all lines **until** the selected text is found.



3. **Include/exclude lines.** Lines between the start and end of the loop that don't contain a line item must be excluded from the extraction. Or rather, all lines that contain a line item have to be included. This is done by adding a **Condition** step within the Repeat step.

1. Select the start of the Repeat step on the Steps pane.
2. Look for something in the data that distinguishes lines with a line item from other lines (or the other way around). Often, a "." or "," appears in prices or totals at the same place in every line item, but not on other lines.
3. Select that data, right-click on it and select **Add Conditional**.



Selecting data - especially something as small as a dot - can be difficult in a PDF file. To make sure that a Condition step checks for certain data: Type the **value** in the **right operand** (in the Step properties pane). Move or resize the selection rectangle in the **data**. Click the **Use selection** button in the **left operand** (in the Step properties pane). When the Condition evaluates to true, the value is found in the selected region.

In the Data Viewer, you will see a green check mark in the left margin next to each included line and an X for other lines.

The screenshot displays a workflow editor on the left, a data viewer in the center, and a 'Condition step' properties pane on the right. A blue callout box points to a 'Condition1' step in the workflow, stating: "A first condition is configured to indicate that only the lines that include an amount of money are extracted".

The data viewer shows a table with columns: UnitPrice, Ordered, B.O., Shipped, and Total. The data is as follows:

UnitPrice	Ordered	B.O.	Shipped	Total
15,98	9	1	8	143,82
104,99	2	1	1	209,98
19,99	12	4	8	239,88
19,99	14	4	10	279,86
44,99	29	7	22	1204,71
199,99	2	1	1	399,98
17,99	19	13	5	323,02
29,99	2	0	2	59,98
14,99	23	12	11	344,77
16,99	5	0	5	84,95
19,99	13	2	11	259,87
44,99	13	12	1	584,87

The 'Condition step' properties pane shows the following configuration:

- Condition list: Condition name
- Left operand: Based on: Position
- Left: 78, Right: 78
- Top offset: 0, Height: 1
- Trim: Both
- Operator: contains
- Right operand: Based on: Value, Value: .

If true, the extraction is performed.
Otherwise, go to the next line
(Goto1)

Product	Description	UnitPrice	Ordered	B.O.	Shipped	Total
HPR4003	Easton SS22 Junior Hocke	15,98	9	1	8	143,92
HPR3001	Bower Valor APX2 Junior H	104,99	2	1	1	209,98
HST2002	CCN Z-22 Junior Wood Hook	19,99	12	4	8	239,88
HML1001	Bower Hockey Cap	19,99	14	4	10	279,86
HST1002	Bower S1 Youth Hockey Sti	44,99	29	7	22	1304,71
HSK5001	Bower Valor APX3 Junior I	199,99	2	1	1	399,98
	Hockey Jersey	17,99	18	13	5	323,82
	E-44 Junior Ho	29,99	2	0	2	59,98
	Cap	14,99	23	12	11	344,77
	Hockey Jersey	16,99	5	0	5	84,95
	Hockey Jersey	19,99	13	2	11	259,87
	Easton SS22 Junior Ice	44,99	13	12	1	584,87
	Hockey Pants					

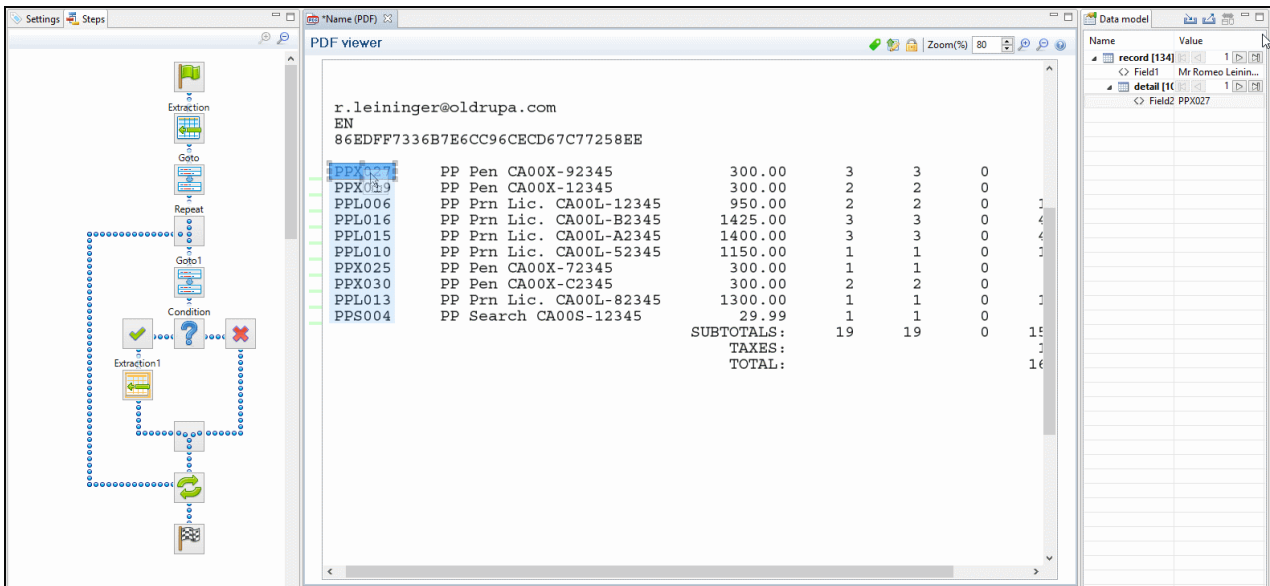
4. (Optional.) **Add an empty detail table** to the Data Model: right-click the Data Model and select **Add a table**. Give the detail table a name.

5. **Extract the data** (see "Adding an extraction" on page 119).

When you drag & drop data on the name of a detail table in the Data Model pane, the data are added to that detail table.

Dropping the data somewhere else on the Data Model pane, or using the contextual menu in the Data Viewer, creates a new detail table, with a default name that you can change later on (see "Renaming a detail table" on page 193).

Name	Value	Total
PPX027	PP Pen CA00X-92345	300.00
PPX019	PP Pen CA00X-12345	300.00
PPL006	PP Prn Lic. CA00L-12345	950.00
PPL016	PP Prn Lic. CA00L-B2345	1425.00
PPL015	PP Prn Lic. CA00L-A2345	1400.00
PPL010	PP Prn Lic. CA00L-52345	1150.00
PPX025	PP Pen CA00X-72345	300.00
PPX030	PP Pen CA00X-C2345	300.00
PPL013	PP Prn Lic. CA00L-82345	1300.00
PPS004	PP Search CA00S-12345	29.99
SUBTOTALS:		19 19 0 15
TAXES:		1
TOTAL:		16



Note

In a PDF or Text file, pieces of data often have a variable size: a product description, for example, may be short and fit on one line, or be long and cover two lines. To learn how to handle this, see "Extracting data of variable length" on the next page.

6. **Extract the sum or totals.** If the record contains sums or totals at the end of the line items list, the end of the Repeat step is a good place to add an Extract step for these data. After the loop step, the cursor position is at the end of line items.
 1. Select the amount or amounts.
 2. Click on the **end** of the Repeat step (🔄) in the Steps panel.
 3. Right-click on the selected data and select **Add Extraction**.

Alternatively, right-click on the end of the Repeat step in the Steps panel and select **Add a Step > Add Extraction**.

The screenshot shows a software interface with three main panes. The left pane displays a flowchart with steps like 'Extraction', 'Goto', 'Repeat', 'Goto1', 'Condition', and 'Extraction1'. The middle pane is a PDF viewer showing a document with a header 'ger@oldrupa.com', a long alphanumeric string '36B7E6CC96CECD67C77258EE', and a table of data. The right pane is a 'Data model' pane showing a hierarchical structure of fields.

Item Description	Price	Quantity	Unit	Other	Total
PP Pen CA00X-92345	300.00	3	3	0	900.00
PP Pen CA00X-12345	300.00	2	2	0	600.00
PP Prn Lic. CA00L-12345	950.00	2	2	0	1900.00
PP Prn Lic. CA00L-B2345	1425.00	3	3	0	4275.00
PP Prn Lic. CA00L-A2345	1400.00	3	3	0	4200.00
PP Prn Lic. CA00L-52345	1150.00	1	1	0	1150.00
PP Pen CA00X-72345	300.00	1	1	0	300.00
PP Pen CA00X-C2345	300.00	2	2	0	600.00
PP Prn Lic. CA00L-82345	1300.00	1	1	0	1300.00
PP Search CA00S-12345	29.99	1	1	0	29.99
SUBTOTALS:		19	19	0	15254.99
TAXES:					1144.12
TOTAL:					16399.11

Tip

This how-to describes in detail how to extract an item description that appears in a variable number of lines: [How to extract multiline items](#).

Extracting data of variable length

In PDF and Text files, transactional data isn't structured uniformly, as in a CSV, database or XML file. Data can be located **anywhere** on a page. Therefore, data are extracted from a certain **region** on the page. However, the data can be spread over **multiple** lines and multiple pages:

- Line items may continue on the next page, separated from the line items on the first page by a line break, a number of empty lines and a letterhead.
- Data may vary in length: a product description for example may or may not fit on one line.

How to exclude lines from an extraction is explained in another topic: "Extracting transactional data" on page 124 (see From a PDF or Text file).

This topic explains a few ways to extract a variable number of lines.

Text file: setting the height to 0

If the variable part in a TXT file is at the end of the record (for example, the body of an email) the height of the region to extract can be set to 0. This instructs the DataMapper to extract all lines starting from a given position in a record until the end of the record, and store them in a single field.

This also works with the `data.extract()` method in a script; see "Examples" on page 271.

Finding a condition

Where it isn't possible to use a setting to extract data of variable length, the key is to find one or more differences between lines that make clear how big the region is from where data needs to be extracted.

Whilst, for example, a product description may expand over two lines, other data - such as the unit price - will never be longer than one line. Either the line above or below the unit price will be empty when the product description covers two lines.

Such a difference can then be used as a condition in a Condition step or a Case in a Multiple Conditions step.

A Condition step, as well as each Case in a Multiple Conditions step, can only check for one condition. To combine conditions, you would need a script.

Using a Condition step or Multiple Conditions step

Using a Condition step ("Condition step" on page 145) or a Multiple Conditions step ("Multiple Conditions step" on page 148) one could determine how big the region is that contains the data that needs to be extracted.

In each of the branches under the Condition or Multiple Conditions step, an Extract step could be added to extract the data from a particular region. The Extract steps could write their data to the same field.

Fields cannot be used twice in one extraction workflow.

Different Extract steps can only write extracted data to the **same** field in the Data Model, if:

- The field name is the same. (See: "Renaming and ordering fields" on page 158.)
- The Extract steps are mutually exclusive. This is the case when they are located in different branches of a Condition step or Multiple Conditions step.
- The option **Append values to current record** is checked in the Step properties pane under Extraction Definition.

Tip

Create and edit the Extract step in the 'true' branch, then right-click the step on the Steps pane, select Copy Step, and paste the step in the 'false' branch. Now you only have to adjust the region from which this Extract step extracts data.

To learn how to configure a Condition step or a Case in a Multiple Conditions step, see "Configuring a Condition step" on page 147.

That second condition check if the Product Number on the next line is empty

A second condition is added under the true branch of the first one to determine which line has a Description field on two lines

Product	Description	UnitPrice	Ordered	B.O.	Shipped	Total
MFR4003	Easton S532 Junior Hockey Elbow Pads	15,98	9	1	8	143,82
MFR3001	Bower Valor APX2 Junior Hockey Gloves	104,99	2	1	1	209,98
HST2002	CCM Z-22 Junior Wood Hockey Stick	19,99	12	4	8	239,88
HAM1001	Bower Hockey Cap	19,99	14	4	10	279,86
HST1002	Bower 51 Youth Hockey Stick	44,99	29	7	22	1304,71
HSK5001	Bower Valor APX3 Junior Hockey skates	199,99	2	1	1	399,98
HAI1003	CCM H22 Hockey Jersey	17,99	18	13	5	323,82
HST1005	Reebok CORE-44 Junior Hockey Stick	29,99	2	0	2	59,98
HAL6001	CCM Hockey Cap	14,99	23	12	11	344,77
HAI1002	CCM Z-35 Hockey Jersey	16,99	5	0	5	84,95
MFR4002	CCM T+2 Youth Hockey Elbow Pads	19,99	13	2	11	259,87
MFR5003	Reebok RKT Junior Hockey Pants	44,99	13	12	1	584,87

If the second condition is true, the extraction is done on the Description fields that extend over two lines

Product	Description	UnitPrice	Ordered	B.O.	Shipped	Total
MFR4003	Easton S532 Junior Hockey Elbow Pads	15,98	9	1	8	143,82
MFR3001	Bower Valor APX2 Junior Hockey Gloves	104,99	2	1	1	209,98
HST2002	CCM Z-22 Junior Wood Hockey Stick	19,99	12	4	8	239,88
HAM1001	Bower Hockey Cap	19,99	14	4	10	279,86
HST1002	Bower 51 Youth Hockey Stick	44,99	29	7	22	1304,71
HSK5001	Bower Valor APX3 Junior Hockey skates	199,99	2	1	1	399,98
HAI1003	CCM H22 Hockey Jersey	17,99	18	13	5	323,82
HST1005	Reebok CORE-44 Junior Hockey Stick	29,99	2	0	2	59,98
HAL6001	CCM Hockey Cap	14,99	23	12	11	344,77
HAI1002	CCM Z-35 Hockey Jersey	16,99	5	0	5	84,95
MFR4002	CCM T+2 Youth Hockey Elbow Pads	19,99	13	2	11	259,87
MFR5003	Reebok RKT Junior Hockey Pants	44,99	13	12	1	584,87

Using a script

A script could also provide a solution when data needs to be extracted from a variable region. This requires using a Javascript-based field.

1. Add a field to an Extract step, preferably by extracting data from one of the possible regions; see "Extracting data" on page 118. To add a field without extracting data, see "JavaScript-based field" on page 156.
2. On the Step properties pane, under Field Definition, select the field and change its **Mode** to **Javascript**.
If the field was created with its Mode set to Location, you will see that the script already contains one line of code to extract data from the original location.
3. Expand the script. Start by doing the check(s) to determine where the data that needs to be extracted is located. Use the `data.extract()` function to extract the data. The parameters that this function expects depend on the data source, see "Examples" on page 271.

Example

The following script extracts data from a certain region in a Text file; let's assume that this region contains the unit price. If the unit price is empty (after trimming any spaces), the product description has to be extracted from two lines; else the product description should be extracted from one line.

```
var s = data.extract(1,7,1,2,"");  
if (s.substring(1,3).trim().length == 0)  
{ data.extract(12,37,0,2,""); } /* extract two lines */  
else { data.extract(12,37,0,1,""); } /* extract one line */
```

The fourth parameter of the `extract()` function contains the height of the region. When working with a Text file, this equals a number of lines.

Tip

With a Text file, the `data.extract()` method accepts **0** as its height parameter. With the height set to 0 it extracts all lines starting from the given position until the end of the record.

Note that this script replicates exactly what can be done in a Condition step. In cases like this, it is recommended to use a Condition step. Only use a script when no steps are sufficient to give the expected result, or when the extraction can be better optimized in a script.

Steps

In the DataMapper, **steps** are part of an extraction workflow (see "Data mapping workflow" on page 113). They contain a specific instruction for the DataMapper, for example to extract data, create a loop, or apply a condition. Some types of steps contain other steps.

Steps are executed sequentially, from top to bottom in an extraction workflow. Inside a Condition step, some steps may be skipped altogether when they are on a particular branch, whereas in a Repeat step - a loop - several steps may be repeated a number of times.

The Preprocessor and Postprocessor steps are special in that the first can be used to modify the incoming data prior to executing the rest of the extraction workflow while the latter can be used to further process the resulting record set after the entire extraction workflow has been executed.

Step types

These are the types of steps that can be added to a data mapping workflow:

- "Preprocessor step" below
- "Extract step" on page 142
- "Repeat step" on page 143
- "Goto step" on page 144
- "Condition step" on page 145
- "Multiple Conditions step" on page 148
- "Action step" on page 149
- "Postprocessor step" on page 150

Preprocessor step


The Preprocessor step allows the application to modify the incoming data prior to executing the rest of the extraction workflow through a number of **preprocessors**. It also lets you define **properties**, to be added to each record or to the data as a whole. A unique ID could be created to be added to each record in the output for integrity checks later on. A time stamp could be

added to create reports. A tag could be added to process certain records differently. A preprocessor could remove certain records altogether.

One example of how a preprocessor could be used is given in a How-to: [Using Preprocessors in DataMapper](#).

Properties


To add a property:

1. Select the **Preprocessor** step on the **Steps** pane.
2. On the **Step properties** pane, under **Properties**, click the **Add** button . See "Properties" on page 217 for an explanation of the settings for properties.

To set the value of a property you can use an Action step (see "Action step" on page 149).

Preprocessors

The Preprocessor step can contain any number of preprocessors. They will be run in sequence before the data is sent to the Data Mapping workflow. To add a preprocessor:

1. Select the **Preprocessor** step on the **Steps** pane.
2. On the **Step properties** pane, under **Preprocessor**, click the **Add** button .
3. Under **Preprocessor definition**, add the script. Preprocessing tasks must be written in JavaScript (see "Using scripts in the DataMapper" on page 255 and "DataMapper Scripts API" on page 252).

Adding an Extract step

To add an Extract step, first select the step on the Steps pane after which to insert the Extract step. Then:

- In the Data Viewer, select some data, right-click that data and choose **Add Extraction**, or drag & drop the data in the Data Model. For more detailed information and instructions, see: "Extracting data" on page 118.
- Alternatively, right-click the Steps pane and select **Add a Step > Add Extraction**. Make the required settings on the Step properties pane.

If an **Extract** step is added within a **Repeat** step, the extracted data are added to a detail table by default; see "Extracting transactional data" on page 124 and "Example " on page 196.

Configuring an Extract step

The names, order, data type and default value of the fields extracted in an Extract step are properties of that Extract step. These and other properties can be edited via the Step properties pane. For an explanation of all the options, see "Settings for location-based fields in a Text file" on page 221.

Fields cannot be used twice in one extraction workflow.

Different Extract steps can only write extracted data to the **same** field in the Data Model, if:

- The field name is the same. (See: "Renaming and ordering fields" on page 158.)
- The Extract steps are mutually exclusive. This is the case when they are located in different branches of a Condition step or Multiple Conditions step.
- The option **Append values to current record** is checked in the Step properties pane under Extraction Definition.

Repeat step

The **Repeat** step is a loop that may run 0 or more times, depending on the condition specified. It is used for the extraction of transactional data; see "Extracting transactional data" on page 124.

Repeat steps do not automatically move the pointer in the source file. Therefore a **Goto** step that moves the cursor is added automatically within the loop to avoid an infinite loop, except in

XML files. When you select a node in an XML file and add a Repeat step on it, the Repeat step will automatically loop over all nodes of the same type on the same level in the XML file.

Adding a Repeat step

To add a Repeat step:

1. On the Steps pane, select the step after which to insert the Condition step.
2. Make sure that the cursor is located where the extraction loop must start. By default the cursor is located at the top of the page or record, but previous steps in the extraction workflow may have moved it down. If necessary, add a **Goto** step (see "Goto step" below).

This step can be skipped when the data source is an XML file.

3. Add the Repeat step:
 - Select data in the line or row where the loop must end, right-click on it and select **Add Repeat**.
 - Right-click the Steps pane and select **Add a Step > Add Repeat**. Make the required settings on the Step properties pane.

Configuring a Repeat step

For information about how to configure the Repeat step, see "Text and PDF Files" on page 233.

How to use it in an extraction workflow is explained in the topic: "Extracting transactional data" on page 124.

Goto step

Although invisible, there is a cursor in the Data Viewer. In an extraction workflow, the cursor starts off at the top-left corner of each record in the source data.

The **Goto** step can move the cursor to a certain location in the current record. The new location can be relative to the top of the record or to the current position.

When the **Goto** step is used within a **Repeat** step, it moves the cursor in each loop of the Repeat step. In this case the new location has to be relative to the current position.

Note

The Goto step isn't used in XML extraction workflows. The DataMapper moves through the file using **Xpath**, a path-like syntax to identify and navigate nodes in an XML document.

Adding a Goto step

To add a Goto step:

- On the Steps pane, select the step after which to insert the Goto step. In the Data Viewer, select some data, right-click that data and choose **Add Goto**, to add a Goto step that moves the cursor to that data.
- Alternatively, right-click the Steps pane and select **Add a Step > Add Goto**. Make the required settings on the Step properties pane.



Configuring a Goto step

For information about how to configure the Goto step, see "Text file" on page 244.

Condition step

A **Condition** step is used when the data extraction must be based on specific criteria. The Condition step splits the extraction workflow into two separate branches, one that is executed when the condition is **true**, the other when it is **false**.

Extract steps can be added to both the 'true' and the 'false' branch (see "Extracting data" on page 118 and "Extracting transactional data" on page 124).

In the Data Viewer pane, icons on the left indicate the result of the evaluation in the Condition step:  when true and  when false.

The screenshot displays a workflow editor with a 'Steps' pane on the left containing steps like Extraction, Goto1, Repeat, Condition, and Goto. The 'Text viewer' window shows a table of items with columns for item name, price, quantity, and a highlighted column of values. The 'Condition step' panel is configured with the following settings:

Item Name	Price	Quantity	Value	Total
Eastown SS32 Junior Hocke	15,98	9	1	143,82
Bower Valor APX2 Junior H	104,99	2	1	209,98
CCN Z-22 Junior Wood Hock	19,99	12	4	239,88
Bower Hockey Cap	19,99	14	4	279,86
Bower S1 Youth Hockey Sti	44,99	29	7	1304,71
Bower Valor APX3 Junior I	199,99	2	1	399,98
CCN H22 Hockey Jersey	17,99	18	13	323,82
Reabuck CORE-44 Junior Ho	29,99	2	0	59,98
CCN Hockey Cap	14,99	23	12	344,77
CCN Z-55 Hockey Jersey	16,99	5	0	84,95
CCN T+2 Youth Hockey Elbo	19,99	13	2	259,87
Reabuck 87K Junior Ice Ho	44,99	13	12	584,87


The 'Condition step' configuration is as follows:

- Condition list: Condition name
- Left operand: Based on: Position; Left: 67; Right: 68; Top offset: 0; Height: 1; Trim: Both
- Operator: is greater than
- Right operand: Based on: Value; Value: 1

Adding a Condition step

To add a Condition step:

- On the Steps pane, select the step after which to insert the Condition step; then, in the Data Viewer, select some data, right-click that data and choose **Add Conditional**. In the **Step properties** pane, you will see that the newly added Condition step checks if the selected **position** (the left operand) **contains** the selected **value** (the right operand). Both operands and the operator can be adjusted. Note that the left operand is by default **trimmed**, meaning that spaces are cut off. Selecting data - especially something as small as a dot - can be difficult in a PDF file. To make sure that a Condition step checks for certain data: Type the **value** in the **right**

operand (in the Step properties pane). Move or resize the selection rectangle in the **data**. Click the **Use selection**  button in the **left operand** (in the Step properties pane). When the Condition evaluates to true, the value is found in the selected region.

- Alternatively, right-click the Steps pane and select **Add a Step > Add Conditional**. Enter the settings for the condition on the Step properties pane.

Configuring a Condition step

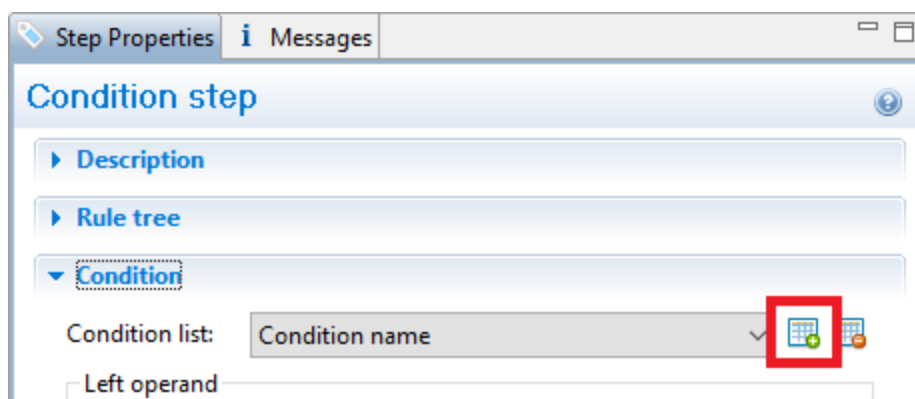
The condition in a Condition step is expressed in a rule or combination of rules. Rules have a left operand, an operand type (for example: contains, is empty) and a right operand. For an overview of all options on the Step properties pane, see "Condition step properties" on page 238.

Inverting a rule

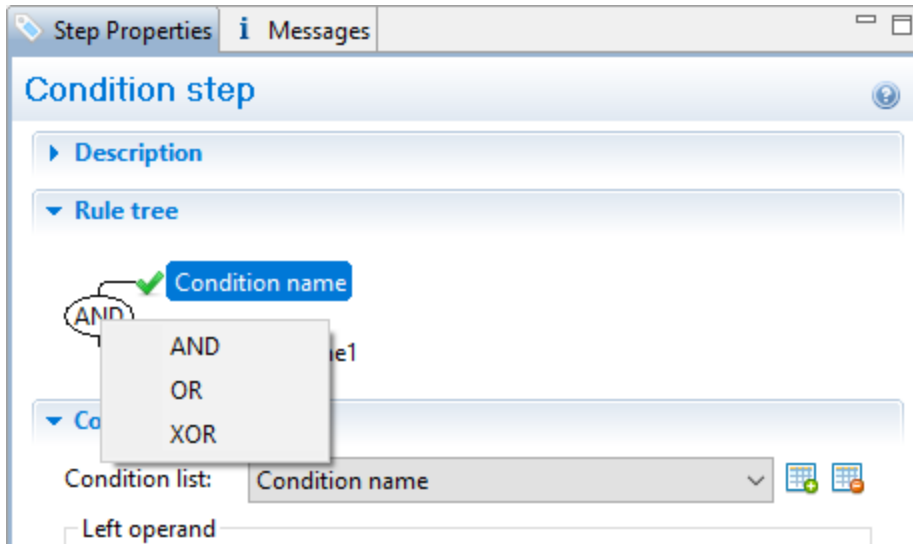
Inverting a rule adds **not** to the operand type. For instance, **is empty** becomes **is not empty**. To invert a rule, check the **Invert condition** option next to Operator under Condition on the Step properties pane.

Combining rules

One rule is already present in a newly added Condition step. To add another rule, click the **Add condition** button under Condition, next to Condition List, on the Step properties pane.



Rules are by default combined with AND. To change the way rules are combined, right-click "AND" in the Rule Tree, on the Step properties pane, and select OR or XOR instead. (XOR means one or the other, but not both.)



Renaming a rule

To rename a rule, double-click its name in the Rule Tree and type a new name.

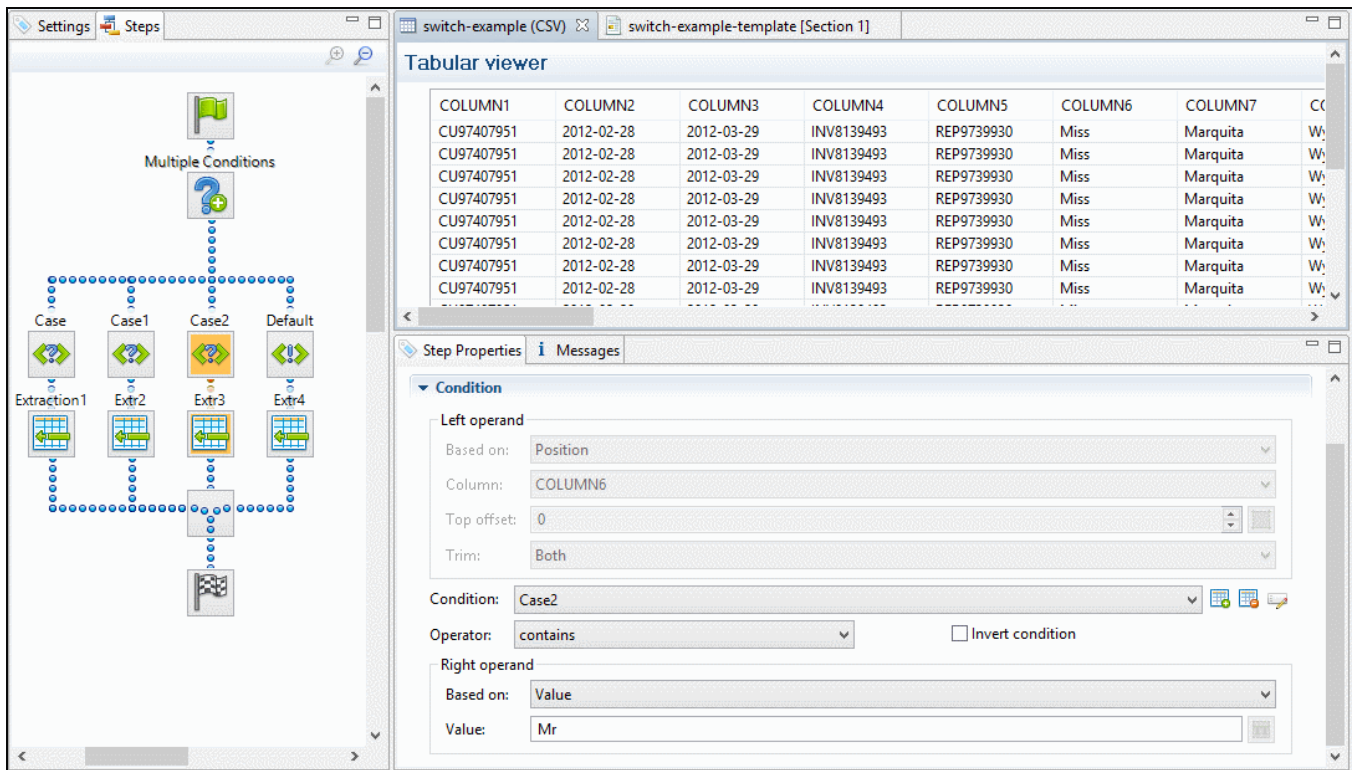
Multiple Conditions step

The **Multiple Conditions** step is useful to avoid the use of nested Condition steps: Condition steps inside other Condition steps.

In a **Multiple Conditions** step, conditions or rather **Cases** are positioned side by side.

Each **Case** condition can lead to an extraction.

Cases are executed from left to right.



Adding a Multiple Conditions step

To add a Multiple Conditions step, right-click the Steps pane and select **Add a Step > Add Multiple Conditions**.

To add a **case**, click the **Add case** button to the right of the **Condition** field in the Step properties pane.

Configuring a Multiple Conditions step

For information about how to configure the Multiple Conditions step, see "Left operand, Right operand" on page 241. The settings for a Case are the same as for a Condition step; see "Condition step properties" on page 238.

Action step

The **Action** step can:

- Execute JavaScript code.
- Set the value for a record property. Record properties are defined in the Preprocessor step; see "Preprocessor step" on page 140.
- Stop the processing of the current record. Normally an extraction workflow is automatically executed on all records in the source data. By stopping the processing of the current record, you can filter records or skip records partially.

The **Action** step can run multiple specific actions one after the other in order.

Adding an Action step

To add an Action step, right-click on the Steps pane and select **Add a Step > Add Action**.

Configuring an Action step


For information about how to configure the Action step, see "Text and PDF Files" on page 226.

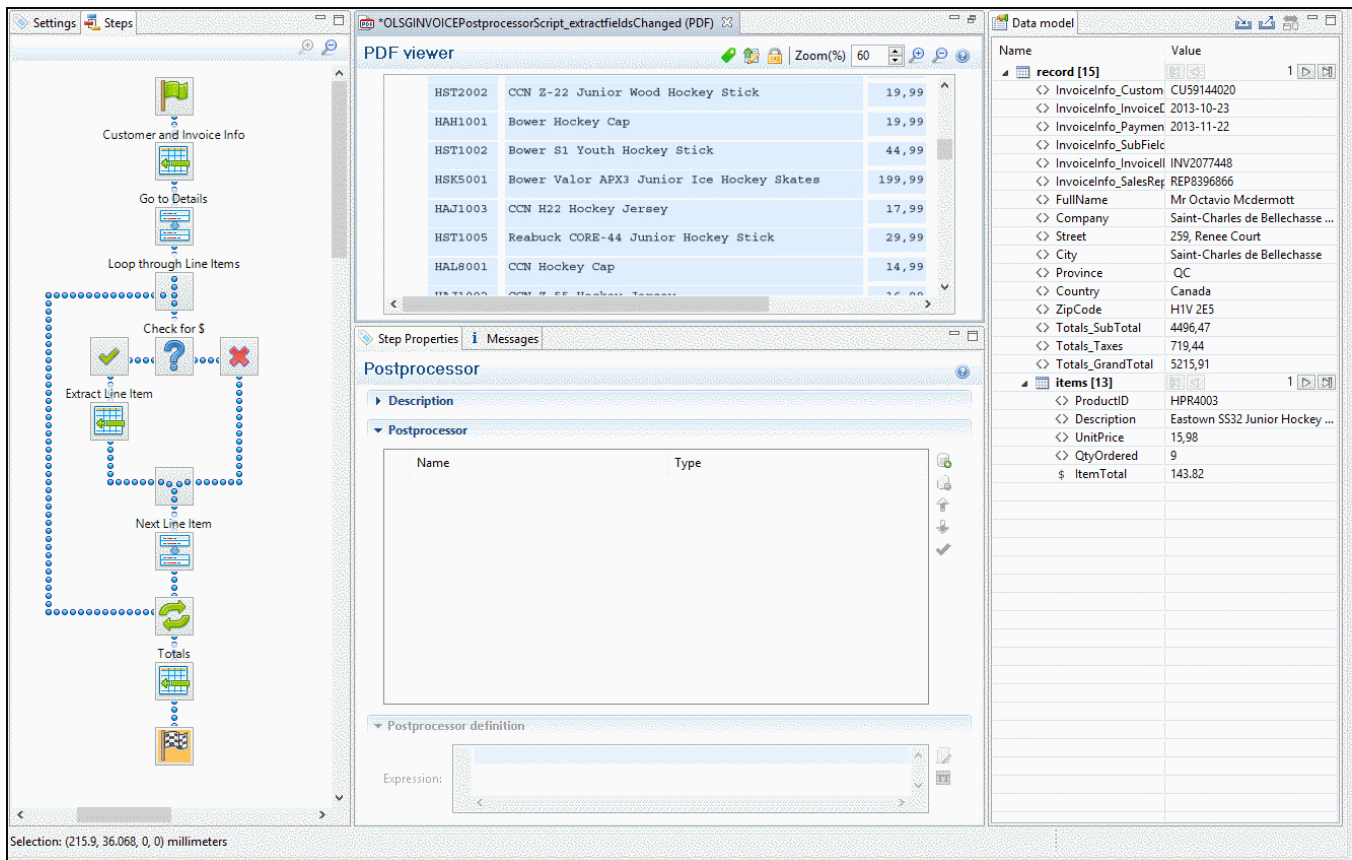
Postprocessor step

The Postprocessor step allows the application to further process the resulting record set after the entire extraction workflow has been executed, using JavaScript. For example, a postprocessor can export all or parts of the data to a CSV file which can then be used to generate daily reports of the Connect Workflow processes that use this data mapping configuration (see "Data mapping configurations" on page 101). A postprocessor could also write the results of the extraction process to a file and immediately upload that file to a Workflow process.

The Postprocessor step can contain any number of postprocessors.

To add a postprocessor:

- Select the **Postprocessor** step on the **Steps** pane.
- On the **Step properties** pane, under **Postprocessor**, click the **Add** button .
- Under **Postprocessor definition**, add the script. Postprocessor tasks must be written in JavaScript (see "Using scripts in the DataMapper" on page 255 and "DataMapper Scripts API" on page 252).



Configuring the Postprocessor step

For an explanation of the settings for postprocessors, see "JavaScript " on page 249.

The Data Model

The Data Model is the structure of records into which extracted data are stored. It contains the names and types of the fields in a record and in its detail tables. A detail table is a field that contains a record set instead of a single value. The Data Model is shown in the Data Model

pane, filled with data from the current record.

Name	Value
record [200]	1
InvNumber	INV1773645
ID	CU63595978
Gender	M
FirstName	Aaron
LastName	Palmer
Title	Mrs
Company	Gigazoom
Address1	73722 Bellgrove Terrace
Country	Canada
State	Ontario
Email	apalmer0@telegraph.co.uk
Phone	1-(560)248-0340
UserName	apalmer0
Password	12u1xgR2
Membership	gold
Language	EN
FavHobby	skate
OrderDate	2016-06-01 12:00 AM
DueDate	2016-07-01 12:00 AM
SubTotal	4148.21
TaxTotal	622.23
Total	4770.44
Products [18]	1
ProdNumber	53674
Description	Thule Crossroad Railing Foot P...
UnitPrice	199.95
Ordered	3
Shipped	3
BackOrder	0
TotalProduct	599.85

The Data Model is not related to the type of data source: whether it is XML, CSV, PDF, Text or a database does not matter. The Data Model is a new structure, designed to contain only the required data.

About records

A **record** is a block of information that may be merged with a template to generate a single *document* (invoice, email, web page...) for a single *recipient*. It is part of the *record set* that is generated by a data mapping configuration.

In each record, data from the data source can be combined with data coming from other sources.



Creating a Data Model

A Data Model is created automatically within each data mapping configuration, but it is empty at the start. To fill it you could use another Data Model (see "Importing/exporting a Data Model" on the next page) or start creating a data mapping workflow (see "Data mapping workflow" on page 113).

To learn how to add and edit fields, see "Fields" on page 155.

Importing/exporting a Data Model

To use a Data Model in another data mapping configuration, or to use it in a Designer template without a data mapping configuration, you have to export that Data Model and import it into a data mapping configuration or template.

Importing and exporting Data Models is done from within the Data model Pane, using the top-right icons  and .

For information about the structure of the exported Data Model file, see "Data Model file structure" on page 177.

When you import a Data Model, it appears in the Data Model pane where you can see all the fields and their types.

You can delete or add fields, or change their type. Once the data model is imported and all the fields are properly set, all you need to do is extract the information from the active data sample (see "Extracting data" on page 118).

Note

- Imported Data Model fields always overwrite existing field properties when the field name is the same (although they will still be part of the same Extract step). Non-existent fields are created automatically with the appropriate field settings. The import is case-insensitive.
- All imported data model fields are tagged with an asterisk (*).

Editing the Data Model

Empty fields and detail tables, added via the Data Model pane, can be edited (renamed, deleted etc.) via the Data Model pane, using the contextual menu that opens when you right-click on a field.

Fields in a Data Model that are actually used in the extraction workflow cannot be edited via the Data Model pane. They are related to a step in the extraction workflow and are edited via the Step properties pane instead.

The order of the fields can also not be changed via the Data Model pane. It is the Extract step that determines the order in which data are extracted, so the order of the fields has to be changed per Extract step.

To learn how to edit fields and change their order, see "Fields" on page 155.

Using the Data Model

The Data Model is what enables you to create personalized templates in the **Designer** module. You can drag & drop fields from the Data Model into the template that you are creating (see "Variable Data" on page 604). For this, you have to have a template and a data mapping configuration open at the same time, or import a Data Model (see "Importing/exporting a Data Model" on the previous page).

The Data Model is reusable, meaning that it can be shared amongst different template layouts and output types.

Different data mapping configurations could use the same Data Model, allowing a template to be populated with data from different sources and formats, without the need to modify the template (see "Importing/exporting a Data Model" on the previous page).

In **Workflow**, when a data mapping configuration is used to extract data from a data source (see "Data mapping configurations" on page 101), the extracted data is stored in a record set that is structured according to the Data Model.

About adding fields and data via Workflow

The Data Model is not extensible outside of the DataMapper. When it is used in Workflow - as part of a data mapping configuration - the contents of its fields can be updated but not its structure.

There are a number of instances however, where fields may need to be added to the data model after the initial data mapping operation has been performed. For instance, you might need to add a cleansed postal address next to the original address, or retrieve a value from a database and add it to the record.

ExtraData field

You can add empty fields in advance to provide space in the Data Model for Workflow to store data. For convenience, one field called **ExtraData** is automatically created at every level of each data record. That means the record itself gets an **ExtraData** field, and each detail table also gets one.

By default the field is not visible in the DataMapper's Data Model, because it is not meant to be filled via an extraction. It can be made visible using the **Show ExtraData Field** icon at the top of the Data Model.

Workflow process

Data can be added to the Data Model in a PlanetPress Connect **Workflow** process as follows:

1. Use an **Execute Data Mapping** task or **Retrieve Items** task to create a record set. On the General tab select **Outputs records in Metadata**.
2. Add a value to a field **in the Metadata** using the **Metadata Fields Management** task. Data added to the **_vger_fld_ExtraData** field on the Document level will appear in the record's ExtraData field, once the records are updated from the Metadata (in the next step).
Other fields have the same prefix: **_vger_fld_**.
3. Update the **record/s** from the Metadata. There are several ways to do this. You could, for example:
 - Use the **Update Data Records** plugin.
 - Add an Output task and check the option **Update records with Metadata**.
 - Select Metadata as the data source in the **Create Preview PDF** plugin.

Note

Many of these actions can also be performed using REST calls.

Please refer to PlanetPress Connect Workflow documentation for more information about the plugins involved.

Fields

Extracted data are stored in fields in the Data Model (see "The Data Model" on page 151). Fields can be present on different levels: on the record level or in a detail table (see "Example " on page 196).

Fields always belong to an Extract step, as can be seen on the Step Properties pane (see "Settings for location-based fields in a Text file" on page 221), but they don't necessarily all contain extracted data.

Location-based fields do: they read data from a certain location in the data source. Other fields may contain the result of a JavaScript (JavaScript-based fields) or the value of a property (property-based fields).

Adding fields

Location-based field

Generally location-based fields are added to a Data Model by extracting data; see "Extracting data" on page 118. Location-based fields in **detail tables** are created by extracting transactional data; see "Extracting transactional data" on page 124.

Alternatively, you can add fields and detail tables directly in the Data Model pane. Right-click anywhere on the Data Model and a contextual menu will appear. Which menu items are available depends on where you've clicked. If you right-click inside the record itself, you can add a field or a detail table. A field will be added at the end with no extraction, while a detail table will be added with no fields inside.

After adding a field or detail table this way, you can drag & drop data into it. Without data it is not accessible via the Step properties pane.

JavaScript-based field

JavaScript-based fields are filled by a script: the script provides a value. Note that the last value attribution to a variable is the one used as the result of the expression.

There is a number of ways to add a Javascript based field.

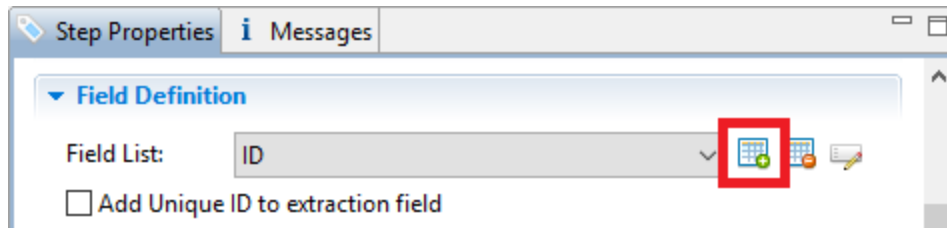
Via the Steps pane

1. Make sure there is **no data selection** in the Data Viewer.
2. Right-click on an Extract step on the Steps pane and select **Add a Step > Add Extract Field**. (To add a new Extract step, select **Add a Step > Add Extraction** first.)
3. On the Step properties pane, under Field Definition, enter the script in the **Expression** field.

Via the Step properties pane

1. Select an Extract step on the Steps pane. (To add a new, empty Extract step, right-click the Steps pane and select **Add a Step > Add Extraction**.)

2. On the Step properties pane, under Field Definition, click the **Add JavaScript Field** button next to the Field List.



3. On the Step properties pane, under Field Definition, enter the script in the **Expression** field.

By changing a field's mode

Alternatively you can change a location-based into a JavaScript-based field.

1. Select the field in the Data Model.
2. On the Step properties pane, under Field Definition, change its **Mode** to JavaScript.
3. Enter the script in the **Expression** field.

Property-based field

A property-based field is filled with the value of a property.

Objects such as the `sourceRecord` and `steps` have a number of predefined properties. (For an explanation of the objects to which the properties belong, see "DataMapper Scripts API" on page 252.)

Custom properties can be added via the Preprocessor step; see "Preprocessor step" on page 140.

A property-based field cannot be added directly. To fill a field with the value of a property, you have to change an existing field's Mode to Properties.

1. Select the field in the Data Model.
2. On the Step properties pane, under Field Definition, change its **Mode** to **Properties**.
3. Select the property from the **Property** drop-down list, or click the button to the right, to open a filter dialog that lets you find a property based on the first few letters that you type.

Another way to add the value of a property to a field is by setting the field's Mode to JavaScript and entering the corresponding property in the Expression field, e.g.

```
data.properties.myProperty;
```

Adding fields dynamically

Outside of the DataMapper the Data Model cannot be changed. It isn't possible to add fields to it when using the data mapping configuration in Workflow. It is however possible to add data to existing fields via Workflow; see "About adding fields and data via Workflow" on page 154.

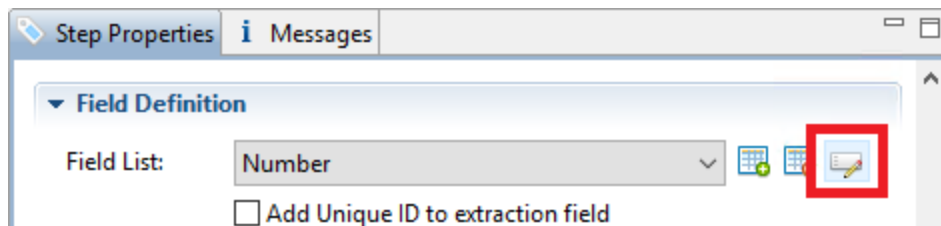
Editing fields

The list of fields that are included in the extraction, the order in which fields are extracted and the data format of each field, are all part of the Extract step's properties. These can be edited via the Step properties pane (see "Settings for location-based fields in a Text file" on page 221).

Renaming and ordering fields

The order and the names of fields in the Data Model can be changed via the properties of the Extract step that they belong to.

1. Select the Extract step that contains the fields that you want to rename. To do this you could click on one of those fields in the Data Model, or on the step in the Steps pane.
2. On the Step properties pane, under Field Definition, click the **Order and rename fields** button.



See "Order and rename fields dialog" on page 224.

Note

Fields cannot have the same name, unless they are on a different level in the record.

If you intend to use the field names as metadata in a Workflow process, do not add spaces to field names, as they are not permitted in metadata field names.

Setting the data type

Fields store extracted data as a String by default. The data type of a field can be changed via the properties of the Extract step that the field belongs to.

1. Select the Extract step that contains the field. You can do this by clicking on the field in the Data Model, or on the step in the Steps pane that contains the field.
2. On the Step properties pane, under Field Definition, set the **Type** to the desired data type. See "Data types" on page 168 for a list of available types.

Changing the type does not only set the data type inside the record. In the case of dates, numbers and currencies, it also means that the DataMapper will expect the data in the data source to be formatted in a certain way. If the actual data doesn't match the format that the DataMapper expects, it cannot interpret the date, number or currency as such. If for example a date in the data source is formatted as "yyyy-mm-dd" but the default format adds the time, the date cannot be read and the DataMapper will stop with an error.

The default format for dates, numbers and currencies can be set in the user preferences ("Datamapper preferences" on page 703), in the data source settings ("Data source settings" on page 115) and per data field (in the Extract step properties, see "Data Format" on page 223).

Setting a default value

You may want to set a default value for a field, in case no extraction can be made. Make sure to set the data type of the field via the step properties (see above). Then right-click the field and select **Default Value**.

The default value must match the selected data type. If the data type of the field is set to Integer, for example, you cannot enter a value of 2,3. A default date must be formatted as a DateTime object ("year-month-day" plus the time stamp, for example: 2012-04-11 12.00 AM); see "Date" on page 171.

Modifying extracted data

To modify extracted data - the contents of a field - you have to write a script. The script can be entered as a Post function in a location-based field or as an Expression in a JavaScript-based field.

Post function

On the Step properties pane, under Field Definition, you can enter a script in the **Post function** field to be run after the extraction. (Click the **Use JavaScript Editor** button to open the [Script Editor](#) dialog if you need more space.)

A Post function script operates directly on the extracted data. Its results replace the extracted data. For example, the Post function script `replace("-", "");` replaces the first dash character that occurs inside the extracted string. The code `toLowerCase();` converts the string to lowercase letters.

Note that the function must be appropriate for the field's data type.

JavaScript Expression

Alternatively you can change a field's **Mode** from Location to Javascript:

1. Select the field in the Data Model.
2. On the Step properties pane, under Field Definition, change its Mode to JavaScript.

You will see that the JavaScript Expression field is not empty; it contains the code that was used to extract data from the location. This code can be used or deleted.

Note

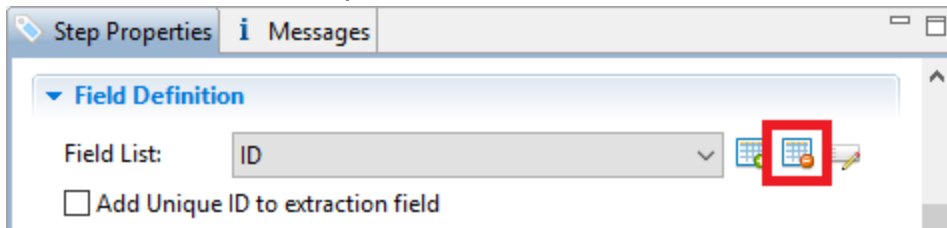
The last value attribution to a variable is the one used as the result of the expression.

Deleting a field

The list of fields that are included in an extraction is one of the properties of an Extract step. To delete a field:

1. Select the field: click on the field in the Data Model, or select the Extract step that contains the field that you want to delete, and in the Step properties pane, under Field Definition, select the field from the Field List.

2. In the Step properties pane, under Field Definition, click the **Remove Extract Field** button next to the Field List drop-down.



Detail tables

A detail table is a field in the Data Model that contains a record set instead of a single value. Detail tables contain transactional data. They are created when an **Extract** step is added within a **Repeat** step; see "Extracting transactional data" on page 124.

In the most basic of transactional communications, a single detail table is sufficient. However, it is possible to create multiple detail tables, as well as nested tables. Detail tables and nested tables are displayed as separate levels in the Data Model (see "The Data Model" on page 151).

Renaming a detail table

Renaming detail tables is especially useful when there are more detail tables in one record, or when a detail table contains another detail table. For this detail table, 'products' would be a better name.

1. On the Data Model pane, click one of the fields in the detail table.
2. On the Step Properties pane, under **Extraction Definition**, in the **Data Table** field, you can find the name of the detail table: **record.detail** by default. Change the **detail** part in that name into something else.

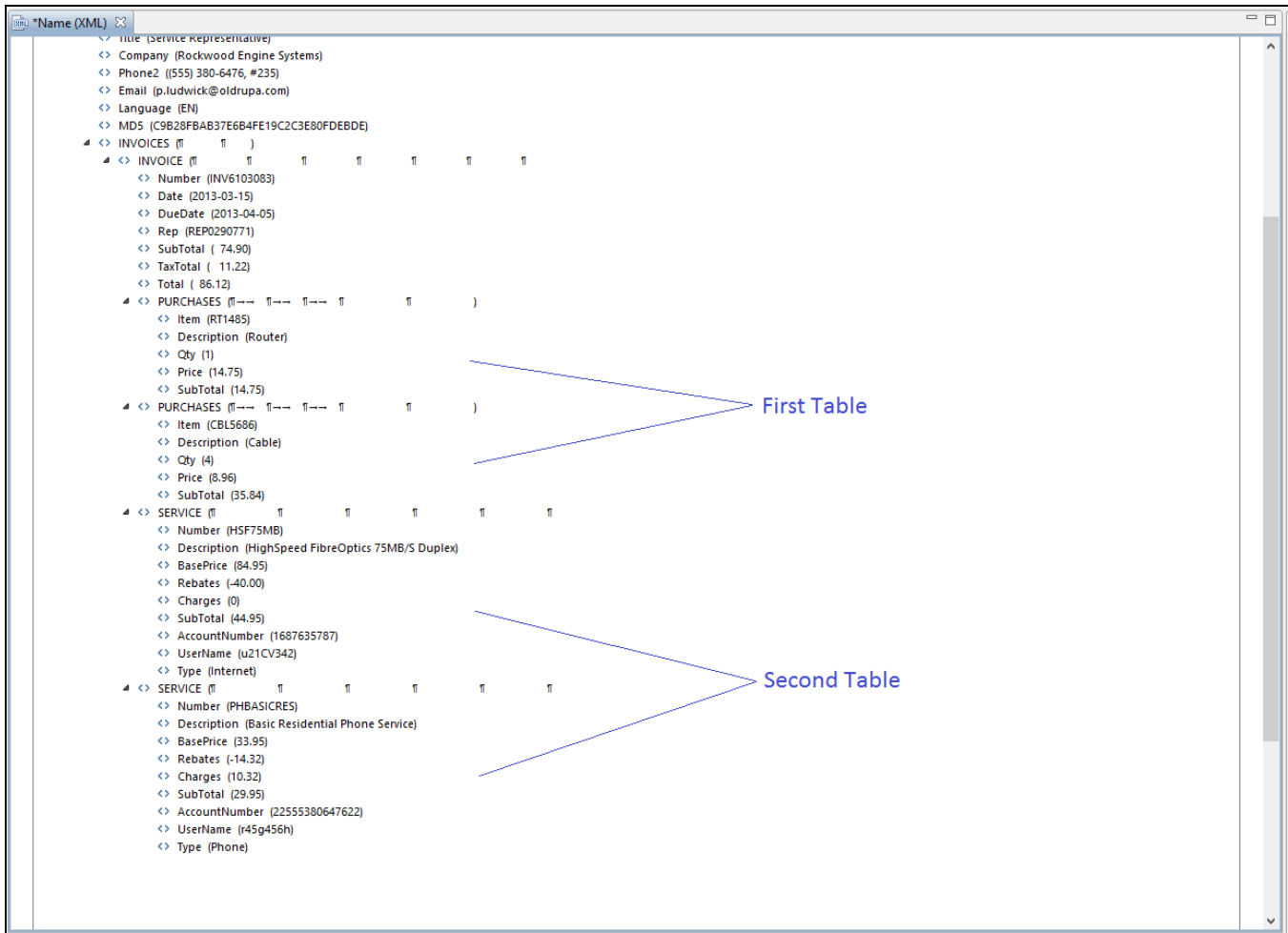
Note

A detail table's name should always begin with '**record.**'

3. Click somewhere else on the Step Properties pane to update the Data Model. You will see the new name appear.

Creating multiple detail tables

Multiple detail tables are useful when more than one type of transactional data is present in the source data, for example purchases (items with a set price, quantity, item number) and services (with a price, frequency, contract end date, etc).



To create more than one detail table, simply extract transactional data in different Repeat steps (see "Extracting transactional data" on page 124).

The best way to do this is to add an empty detail table (right-click the Data Model, select **Add a table** and give the detail table a name) and drop the data on the name of that detail table.

Else the extracted fields will all be added to one new detail table with a default name at first, and you will have to rename the detail table created in each Extract step to pull the detail tables apart (see "Renaming a detail table" on the previous page).

The screenshot displays a software interface with three main panels. The left panel shows a workflow with 'Extraction' and 'Extraction1' steps. The middle panel shows an XML tree structure for an invoice, including fields like Phone2, Email, Language, MDS, and INVOICES. The right panel shows a 'Data model' table with columns for Name and Value, listing various fields and their corresponding values.

XML Structure:

```

<> Phone2 ((555) 702-5520, #665)
<> Email (a.tincher@oldrupa.com)
<> Language (EN)
<> MDS (BE9479B978EE0AE3970837291B7105C8)
<> INVOICES (⌈ ⌋)
  <> INVOICE (⌈ ⌋)
    <> Number (INV8559825)
    <> Date (2013-03-15)
    <> DueDate (2013-04-05)
    <> Rep (REP8800122)
    <> SubTotal ( 125.96)
    <> TaxTotal ( 18.86)
    <> Total ( 144.82)
    <> PURCHASES (⌈--- ⌋--- ⌋--- ⌋)
      <> Item (RT1485)
      <> Description (Router)
      <> Qty (1)
      <> Price (14.75)
      <> SubTotal (14.75)
      <> PURCHASES (⌈--- ⌋--- ⌋--- ⌋)
        <> Item (CBL5686)
        <> Description (Cable)
        <> Qty (3)
        <> Price (8.96)
        <> SubTotal (26.88)
        <> PURCHASES (⌈--- ⌋--- ⌋--- ⌋)
          <> Item (FST5678)
          <> Description (Fasteners)
          <> Qty (50)
          <> Price (0.25)
          <> SubTotal (12.5)
    <> <SRVICF (⌈ ⌋)
  
```

Data Model Table:

Name	Value
record [3]	
abc FULLNAME	Amie Tincher
abc ID	CU44643835
abc Gender	Miss
abc LastName	Tincher
abc FirstName	Amie
abc Address1	356, Chambers Street
abc Address2	P.O. Box 6049
abc City	St. Mary's
abc State	ON
abc Country	CA
abc ZipCode	L6J 5L1
abc Title	Driver/Sales Workers
abc Company	St. Mary's Pest Control Ltd
abc Phone2	(555) 702-5520, #665
abc Email	a.tincher@oldrupa.com
abc Language	EN
abc MDS	BE9479B978EE0AE397083...
abc Number	INV8559825
abc Date	2013-03-15
abc DueDate	2013-04-05
abc Rep	REP8800122
abc SubTotal	125.96
abc TaxTotal	18.86
abc Total	144.82

Step Properties:

Extract Step

Description

Extraction Definition

Data Table: record

Append values to current record

The screenshot displays a software interface with three main panels. The left panel shows a workflow with 'Extraction', 'Extraction1', 'Repeat', 'Extraction2', 'Repeat1', and 'Extraction3' steps. The middle panel shows an XML tree structure for a service, including fields like Sub-Total, PURCHASES, SERVICE, and details like Number, Description, BasePrice, Rebates, Charges, AccountNumber, and UserName. The right panel shows a 'Data model' table with columns for Name and Value, listing various fields and their corresponding values.

XML Structure:

```

<> Sub-Total (14.75)
<> PURCHASES (⌈--- ⌋--- ⌋--- ⌋)
  <> Item (CBL5686)
  <> Description (Cable)
  <> Qty (3)
  <> Price (8.96)
  <> SubTotal (26.88)
  <> PURCHASES (⌈--- ⌋--- ⌋--- ⌋)
    <> Item (FST5678)
    <> Description (Fasteners)
    <> Qty (50)
    <> Price (0.25)
    <> SubTotal (12.5)
  <> SERVICE (⌈ ⌋)
    <> Number (HSF75MB)
    <> Description (HighSpeed FibreOptics 75MB/S Duplex)
    <> BasePrice (84.95)
    <> Rebates (-40.00)
    <> Charges (0)
    <> SubTotal (44.95)
    <> AccountNumber (1687635787)
    <> UserName (u21CV342)
    <> Type (Internet)
  <> SERVICE (⌈ ⌋)
    <> Number (PHBASICSRES)
    <> Description (Basic Residential Phone Service)
    <> BasePrice (33.95)
    <> Rebates (-14.32)
    <> Charges (22.32)
    <> SubTotal (41.95)
    <> AccountNumber (22555705552022)
    <> UserName (w457478)
  
```

Data Model Table:

Name	Value
record [3]	
abc FULLNAME	Amie Tincher
abc ID	CU44643835
abc Gender	Miss
abc LastName	Tincher
abc FirstName	Amie
abc Address1	356, Chambers Street
abc Address2	P.O. Box 6049
abc City	St. Mary's
abc State	ON
abc Country	CA
abc ZipCode	L6J 5L1
abc Title	Driver/Sales Workers
abc Company	St. Mary's Pest Control Ltd
abc Phone2	(555) 702-5520, #665
abc Email	a.tincher@oldrupa.com
abc Language	EN
abc MDS	BE9479B978EE0AE397083...
abc Number	INV8559825
abc Date	2013-03-15
abc DueDate	2013-04-05
abc Rep	REP8800122
abc SubTotal	125.96
abc TaxTotal	18.86
abc Total	144.82
detail [6]	
abc Item	RT1485
abc Description	Router
abc Qty	1
abc Price	14.75
abc SubTotal2	14.75
abc Number2	
abc Description2	
abc BasePrice	
abc Rebates	
abc Charges	
abc SubTotal3	
abc AccountNum	
abc UserName	
abc Type	

Step Properties:

Extract Step

Description

Extraction Definition

Data Table: record.detail

Append values to current record

Nested detail tables

Nested detail tables are used to extract transactional data that are relative to other data. They are created just like multiple detail tables, with two differences:

- For the tables to be actually nested, the **Repeat** step and its **Extract** step that extract the nested transactional data must be located **within** the **Repeat** step that extracts data to a detail table.
- In their name, the dot notation (record.services) must contain one extra level (record.services.charges).

Note

Using nested detail tables in the **Designer** module requires scripting, as described in this How-to: [Cloning your way through nested tables](#).

Example

An XML source file lists the services of a multi-service provider: Internet, Cable, Home Phone, Mobile. Each service in turn lists a number of "charges", being service prices and rebates, and

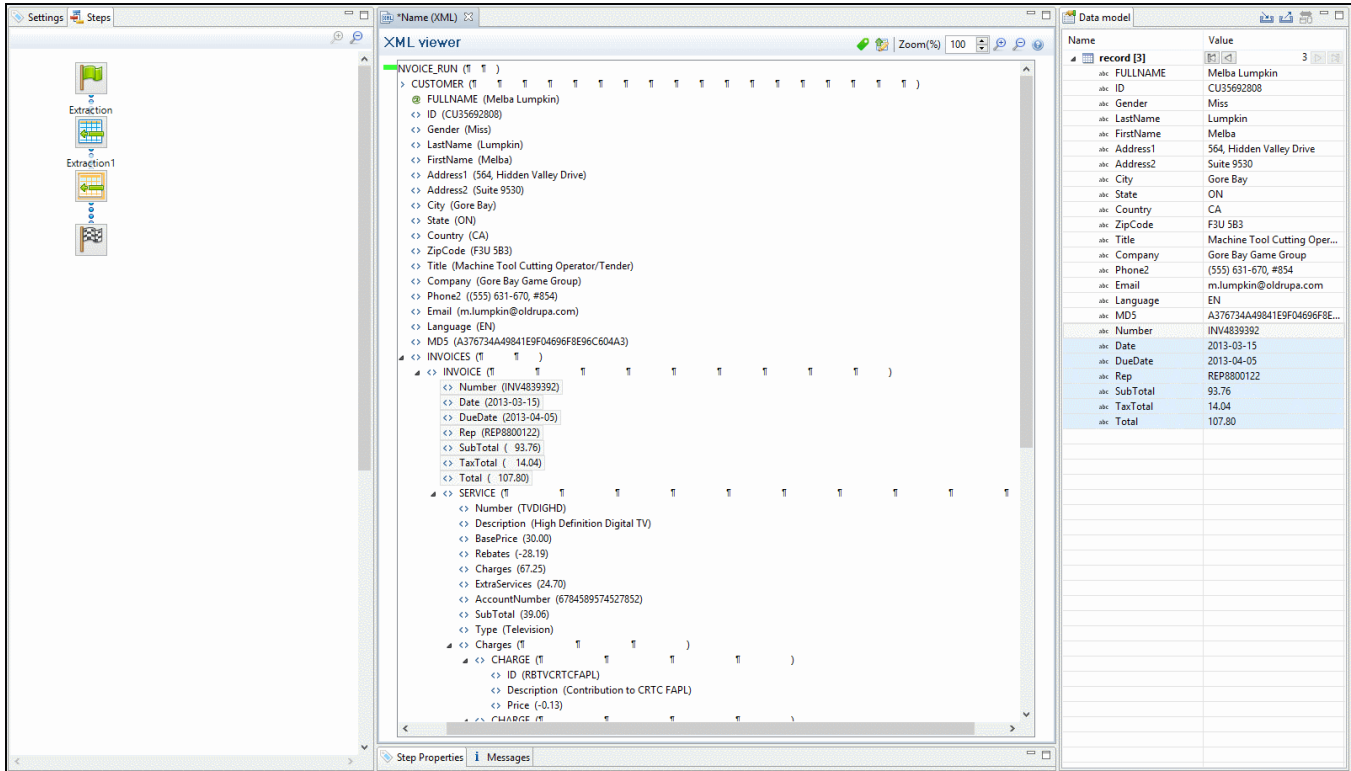
a number of "details" such as movie rentals or long distance calls.

XML viewer

- <> Rep (REP8800122)
- <> SubTotal (93.76)
- <> TaxTotal (14.04)
- <> Total (107.80)
- ▲ <> SERVICE (¶ ¶ ¶ ¶)
 - <> Number (TVDIGHD)
 - <> Description (High Definition Digital TV)
 - <> BasePrice (30.00)
 - <> Rebates (-28.19)
 - <> Charges (67.25)
 - <> ExtraServices (24.70)
 - <> AccountNumber (6784589574527852)
 - <> SubTotal (39.06)
 - <> Type (Television)
- ▲ <> Charges (¶ ¶ ¶)
 - ▲ <> CHARGE (¶ ¶ ¶ ¶)
 - <> ID (RBTVCRTCFAPL)
 - <> Description (Contribution to CRTC FAPL)
 - <> Price (-0.13)
 - ▲ <> CHARGE (¶ ¶ ¶ ¶)
 - <> ID (RBTVPROG)
 - <> Description (PROMO: Programming Credit)
 - <> Price (-14.20)
 - ▲ <> CHARGE (¶ ¶ ¶ ¶)
 - <> ID (RBTVHDBOX)
 - <> Description (PROMO: Free HD Receiver Rental)
 - <> Price (-13.86)
- ▲ <> DETAILS (¶ ¶ ¶ ¶)
 - ▲ <> DETAIL (¶ ¶ ¶ ¶)
 - <> CODE (MVHDLOC)
 - <> ID (55123)
 - <> Description (Gangster Squad HD)
 - <> TimeStamp (2013-02-16 19:24:33)
 - <> Price (5.95)
 - ▲ <> DETAIL (¶ ¶ ¶ ¶)
 - <> CODE (MVHDLOC)
 - <> ID (55234)
 - <> Description (Indiana Jones: Raiders of the Lost Ark)
 - <> TimeStamp (2013-02-23 18:02:46)
 - <> Price (2.95)
 - ▲ <> DETAIL (¶ ¶ ¶ ¶)
 - <> CODE (MVHDLOC)
 - <> ID (55345)
 - <> Description (Indiana Jones and the Temple of Doom)

Data to be Nested

The services can be extracted to a detail table called **record.services**.



The "charges" and "details" can be extracted to two nested detail tables.

The screenshot shows a workflow editor on the left with steps: Extraction, Extraction1, Repeat, Extraction2, and a final output step. The XML viewer in the center displays the following structure:

```

< INVOICE (INV4839392)
  < Number (INV4839392)
  < Date (2013-03-15)
  < DueDate (2013-04-05)
  < Rep (REP8800122)
  < SubTotal ( 93.76)
  < TaxTotal ( 14.04)
  < Total ( 107.80)
  < SERVICE (TVDIGHD)
    < Number (TVDIGHD)
    < Description (High Definition Digital TV)
    < BasePrice (30.00)
    < Rebates (-28.19)
    < Charges (67.25)
    < ExtraServices (24.70)
    < AccountNumber (6784589574527852)
    < SubTotal (39.06)
    < Type (Television)
    < Charges ( )
      < CHARGE ( )
        < ID (RBTVCRTCFAPL)
        < Description (Contribution to CRTFC FAPL)
        < Price (-0.13)
      < CHARGE ( )
        < ID (RBTVPROG)
        < Description (PROMO: Programming Credit)
        < Price (-14.20)
      < CHARGE ( )
        < ID (RBTVHDBOX)
        < Description (PROMO: Free HD Receiver Rental)
        < Price (-13.86)
    < DETAILS ( )
      < DETAIL ( )
        < CODE (MVHDLOC)
        < ID (55123)
        < Description (Gangster Squad HD)
        < Timestamp (2013-02-16 19:24:33)
        < Price (5.95)
      < DETAIL ( )
        < CODE (MVHDLOC)
        < ID (55234)
        < Description (Indiana Jones: Raiders of the Lost Ark)
        < Price (7.95)
  
```

The Data model table on the right shows the following data:

Name	Value
record [3]	3
akc FULLNAME	Melba Lumpkin
akc ID	CU35692808
akc Gender	Miss
akc LastName	Lumpkin
akc FirstName	Melba
akc Address1	564, Hidden Valley Drive
akc Address2	Suite 9530
akc City	Gore Bay
akc State	ON
akc Country	CA
akc ZipCode	F3U 5B3
akc Title	Machine Tool Cutting Oper...
akc Company	Gore Bay Game Group
akc Phone2	(555) 631-670, #854
akc Email	m.lumpkin@oldrupa.com
akc Language	EN
akc MD5	A376734A49841E9F04669F8E...
akc Number	INV4839392
akc Date	2013-03-15
akc DueDate	2013-04-05
akc Rep	REP8800122
akc SubTotal	93.76
akc TaxTotal	14.04
akc Total	107.80
detail [1]	1
akc Number2	TVDIGHD
akc Description	High Definition Digital TV
akc BasePrice	30.00
akc Rebates	-28.19
akc Charges	67.25
akc ExtraServices	24.70
akc AccountNumber	6784589574527852
akc SubTotal2	39.06
akc Type	Television

The nested tables can be called **record.services.charges** and **record.services.details**.

The screenshot shows a more complex workflow with steps: Extraction, Extraction1, Repeat, Extraction2, Repeat1, Extraction3, Repeat2, Extraction4, and a final output step. The XML viewer displays the following structure:

```

< SERVICE (TVDIGHD)
  < Number (TVDIGHD)
  < Description (High Definition Digital TV)
  < BasePrice (30.00)
  < Rebates (-28.19)
  < Charges (67.25)
  < ExtraServices (24.70)
  < AccountNumber (6784589574527852)
  < SubTotal (39.06)
  < Type (Television)
  < Charges ( )
    < CHARGE ( )
      < ID (RBTVCRTCFAPL)
      < Description (Contribution to CRTFC FAPL)
      < Price (-0.13)
    < CHARGE ( )
      < ID (RBTVPROG)
      < Description (PROMO: Programming Credit)
      < Price (-14.20)
    < CHARGE ( )
      < ID (RBTVHDBOX)
      < Description (PROMO: Free HD Receiver Rental)
      < Price (-13.86)
  < DETAILS ( )
    < DETAIL ( )
      < CODE (MVHDLOC)
  
```

The Data model table on the right shows the following data:

Name	Value
record [3]	3
akc FULLNAME	Melba Lumpkin
akc ID	CU35692808
akc Gender	Miss
akc LastName	Lumpkin
akc FirstName	Melba
akc Address1	564, Hidden Valley Drive
akc Address2	Suite 9530
akc City	Gore Bay
akc State	ON
akc Country	CA
akc ZipCode	F3U 5B3
akc Title	Machine Tool Cutting Oper...
akc Company	Gore Bay Game Group
akc Phone2	(555) 631-670, #854
akc Email	m.lumpkin@oldrupa.com
akc Language	EN
akc MD5	A376734A49841E9F04669F8E...
akc Number	INV4839392
akc Date	2013-03-15
akc DueDate	2013-04-05
akc Rep	REP8800122
akc SubTotal	93.76
akc TaxTotal	14.04
akc Total	107.80
details [10]	1
akc Number2	TVDIGHD
akc Description	High Definition Digital TV
akc BasePrice	30.00
akc Rebates	-28.19
akc Charges	67.25
akc ExtraServices	24.70
akc AccountNumber	6784589574527852
akc SubTotal2	39.06
akc Type	Television
akc ID2	
akc Description2	
akc Price	
akc CODE	
akc ID3	
akc Description3	
akc Timestamp	
akc Price2	

The Preprocessor configuration window is also visible, showing a table for Fixed automation properties:

Name	Scope	Type	Default Value

Now one "charges" table and one "details" table are created for each row in the "services" table.

Data types

By default the data type of extracted data is a String, but each field in the Data Model can be set to contain another data type.

To do this:

1. In the **Data Model**, select a field.
2. On the **Step properties** pane, under **Field Definition** choose a data type from the **Type** drop-down.

Changing the type does not only set the data type inside the record. In the case of dates, numbers and currencies, it also means that the DataMapper will expect the data in the data source to be formatted in a certain way. If the actual data doesn't match the format that the DataMapper expects, it cannot interpret the date, number or currency as such. If for example a date in the data source is formatted as "yyyy-mm-dd" but the default format adds the time, the date cannot be read and the DataMapper will stop with an error.

The default format for dates, numbers and currencies can be set in the user preferences ("Datamapper preferences" on page 703), in the data source settings ("Data source settings" on page 115) and per data field (in the Extract step properties, see "Data Format" on page 223).

Note

Data format settings tell the DataMapper how certain types of data are formatted **in the data source**. They don't determine how these data are formatted in the Data Model or in a template. In the Data Model, data are converted to the native data type. Dates, for example, are converted to a DateTime object in the Data Model, and will always be shown as "year-month-day" plus the time stamp, for example: 2012-04-11 12.00 AM.

The following data types are available in PlanetPress Connect.

- "Boolean" on the next page
- "String" on page 176

- "HTMLString" on page 175
- "Integer" on page 175
- "Float" on page 174
- "Currency" on the facing page
- "Date" on page 171
- "Object" on page 176

Note

The Object data type is only available in the DataMapper module. It can be used for properties in the Preprocessor step, but not for fields in the Data Model.

Boolean

Booleans are a simple true/false data type often used in conditions and comparisons.

Defining Boolean values

- **Preprocessor:**
 - In the **Step properties** pane, under **Properties**, add or select a field.
 - Specify the **Type** as **Boolean** and set a default value of either `true` or `false`, followed by a semicolon.
- **Extraction:**
 - In the **Data Model**, select a field.
 - On the **Step properties** pane, under **Field Definition** set the **Type** to **Boolean**. The field value must be `true` or `false`.
- **JavaScript Expression:** Set the desired value to either `true` or `false`. Example:

```
record.fields["isCanadian"] = true;
```

Note

The value must be all in lowercase: `true`, `false`. Any variation in case (True, TRUE) will not work.

Boolean expressions

Boolean values can also be set using an expression of which the result is true or false. This is done using operators and comparisons.

Example: `record.fields["isCanadian"] = (extract("Country") == "CA");`

For more information on JavaScript comparison and logical operators, please see w3schools.com or developer.mozilla.org.

Currency

The Currency data type is a signed, numeric, fixed-point 64-bit number with 4 decimals. Values range from -922 337 203 685 477.5808 to 922 337 203 685 477.5808. This data type is routinely used for financial calculations: it is as precise as integers.

Defining Currency values

- **Preprocessor:**
 - In the **Step properties** pane, under **Properties**, add or select a field.
 - Specify the **Type** as **Currency** and set a default value as a number with up to 4 decimal points, followed by a semicolon; such as `546513.8798;`
- **Extraction:**
 - In the **Data Model**, select a field.
 - On the **Step properties** pane, under **Field Definition** set the **Type** to **Currency**.
 - Under **Data Format**, specify how the value is formatted in the data source (see [Extract Step](#); for the default format settings, see "Data source settings" on page 115).
The field value will be extracted and treated as a Float.
- **JavaScript Expression:** Set the desired value to any Float value. Example:
`record.fields["PreciseTaxSubtotal"] = 27.13465;`

Note

While Currency values can be set to up to 4 significant digits, only 2 are displayed on screen.

Building Currency values

Currency values can be the result of direct attribution or mathematical operations just like Integer values (see "Integer" on page 175).

Date

Dates are values that represent a specific point in time, precise up to the second. They can also be referred to as `datetime` values. While dates are displayed using the system's regional settings, in reality they are stored unformatted.

Note

The **Date** property is stored in **Connect** database with zero time zone offset, which makes it possible to convert the time correctly in any location. PlanetPress Workflow, however, shows the date/time as it is stored database (with 0 time zone offset). This is expected behavior for the moment and the zone offset must be calculated manually in PlanetPress Workflow.

Extracting dates

To extract data and have that data interpreted as a Date, set the type of the respective field to Date:

1. Select the field in the data model.
2. On the **Step properties** pane, under **Field Definition**, specify the **Type** as **Date**.
3. Make sure that the date in the data source is formatted in a way that matches the expectations of the DataMapper. If the date doesn't match the format that the DataMapper expects, it cannot be interpreted as a date. For example, if a date in the data source is formatted as "yyyy-mm-dd" but the DataMapper expects a time as well, the date cannot be read and the DataMapper will stop with an error.

The expected date format can be set in three places:

- In the user preferences ("Datamapper preferences" on page 703).
- In the data source settings ("Data source settings" on page 115).
- In the field properties: on the **Step properties** pane, under **Data Format**, specify the **Date/Time Format**.

For the letters and patterns that you can use in a date format, see "Defining a date/time format" below.

Data format settings tell the DataMapper how certain types of data are formatted **in the data source**. They don't determine how these data are formatted in the Data Model or in a template. In the Data Model, data are converted to the native data type. Dates, for example, are converted to a DateTime object in the Data Model, and will always be shown as "year-month-day" plus the time stamp, for example: 2012-04-11 12.00 AM..

Defining a date/time format

A date format is a mask representing the order and meaning of each digit in the raw data, as well as the date/time separators. The mask uses several predefined markers to parse the contents of the raw data. Here is a list of markers that are available in the DataMapper:

- **yy**: Numeric representation of the Year when it is written out with only 2 digits (i.e. 13)
- **yyyy**: Numeric representation of the Year when it is written out with 4 digits (i.e. 2013)
- **M**: Short version of the month name (i.e. Jan, Aug). These values are based on the current regional settings.
- **MM**: Long version of the month name (i.e. January, August). These values are based on the current regional settings.
- **mm**: Numeric representation of the month (i.e. 1, 09, 12)
- **D**: Short version of the weekday name (i.e. Mon, Wed). These values are based on the current regional settings.
- **DD**: Long version of the weekday name (i.e. Monday, Wednesday). These values are based on the current regional settings.
- **dd**: Numeric representation of the day of the month (i.e. 1, 09, 22)
- **hh**: Numeric representation of the hours
- **nn**: Numeric representation of the minutes
- **ss**: Numeric representation of the seconds
- **ms**: Numeric representation of the milliseconds.
- **ap**: AM/PM string.
- In addition, any constant character can be included in the mask, usually to indicate date/time separators (i.e. / - :). If one of those characters happens to be one of the reserved characters listed above, it must be escaped using the \ symbol.

Note

The markers that can be used when **extracting** dates are different from those that are used to **display** dates in a template (see the Designer's "Date and time patterns" on page 918).

Examples of masks

Value in raw data	Mask to use
June 25, 2013	MM dd, YYYY
06/25/13	mm/dd/yy
2013.06.25	yyyy.mm.dd
2013-06-25 07:31 PM	yyyy-mm-dd hh:nn ap
2013-06-25 19:31:14.1206	yyyy-mm-dd hh:nn:ss.ms
Tuesday, June 25, 2013 @ 7h31PM	DD, MM dd, yyyy @ hh\hnnap

Entering a date using JavaScript

In several places in the DataMapper, Date values can be set through a JavaScript. For example:

- In a **field** in the Data Model. To do this, go to the Steps pane and select an Extract step. Then, on the Step properties pane, under Field Definition click the **Add JavaScript Field** button (next to the Field List drop-down). Type the JavaScript in the **Expression** field. (To rename the field, click the Order and rename fields button.)
- In a **Preprocessor property**. To do this, go to the Steps pane and select the Preprocessor step. Then, on the Step properties pane, under Properties add a property, specify its Type as Date and put the JavaScript in the Default Value field.

The use of the JavaScript Date() object is necessary when creating dates through a JavaScript expression. For more information, see [w3schools - JavaScript Dates](#) and [w3schools - Date Object](#).

Example

The following script creates a date that is the current date + 30 days:

```
function addDays(date, days) {
    var result = new Date(date);
    result.setDate(result.getDate() + days);
    return result;
}
addDays(new Date(), 30);
```

Float

Floats are signed, numeric, floating-point numbers whose value has 15-16 significant digits. Floats are routinely used for calculations. Note that Float values can only have up to 3 decimals. They are inherently imprecise: their accuracy varies according to the number of significant digits being requested.

The Currency data type can have up to 4 decimals; see "Currency" on page 170.

Defining Float values

- **Preprocessor:**
 - In the **Step properties** pane, under **Properties**, add or select a field.
 - Specify the **Type** as **Float** and set a default value as a number with decimal points, followed by a semicolon; for example `546513.879;`.
- **Extraction:**
 - In the **Data Model**, select a field.
 - On the **Step properties** pane, under **Field Definition** set the **Type** to **Float**. The field value will be extracted and treated as a Float.
- **JavaScript Expression:** Set the desired value to any Float value.
Example: `record.fields["PreciseTaxSubtotal"] = 27.134;`

Building Float values

Float values can be the result of direct attribution or mathematical operations just like Integer values (see "Integer" on the next page).

HTMLString

HTMLStrings contain textual data that includes HTML markup. They are essentially the same as String values except in cases where the HTML markup can be interpreted.

Example: Assume that a field has the value `He said WOW!`. If the data type is String and the value is placed in a template, it will display exactly as "He said `WOW!`" (without the quotes). If the data type is HTMLString, it will display as "He said **WOW!**" (again, without the quotes).

Considering this is the only difference, for more information on how to create and use HTML String values, see "String" on the facing page.

Integer

Integers are signed, numeric, whole 64bit numbers whose values range from $-(2^{63})$ to (2^{63}) . Integers are the numerals with the highest precision (and the fastest processing speed) of all, since they are never rounded.

Defining Integer values

- **Preprocessor:**
 - In the **Step properties** pane, under **Properties**, add or select a field.
 - Specify the **Type** as **Integer** and set a default value as a number, such as 42.
- **Extraction:**
 - In the **Data Model**, select a field.
 - On the **Step properties** pane, under **Field Definition** set the **Type** to **Integer**. The field value will be extracted and treated as an integer.
- **JavaScript Expression:** Set the desired value to any Integer value. Example:

```
record.fields["AnswerToEverything"] = 42;
```

Building Integer Values

Integers can be set through a few methods, all of which result into an actual integer result.

- **Direct attribution:** Assign an integer value directly, such as 42, 99593463712 or

```
data.extract("TotalOrdered");
```

- **Mathematical operations:** Assign the result of any mathematical operation. For example: `22+51`, `3*6`, `10/5` or `sourceRecord.property.SubTotal`. For more information on mathematics in JavaScript, see [w3Schools - Mathematical Operators](#). For more advanced mathematical functions, see [w3schools - Math Object](#).

Note

When adding numbers that are not integers, for instance `4.5 + 1.2`, a round towards zero rounding is applied after the operation was made. In the previous example, the result, `5.7`, is rounded to `5`. In another example, `-1.5 - 1` results in `-2`

Object

Objects holds addresses that refer to objects. You can assign any reference type (string, array, class, or interface) to an Object variable. An Object variable can also refer to data of any value type (numeric, Boolean, Char, Date, structure, or enumeration).

Defining Object values

- **Preprocessor:**
 - In the **Step properties** pane, under **Properties**, add or select a field.
 - Specify the **Type** as **Object** and set a default value as a semi-colon.

String

Strings contain textual data. Strings do not have any specific meaning, which is to say that their contents are never interpreted in any way.

Defining String values

- **Preprocessor:**
 - In the **Step properties** pane, under **Properties**, add or select a field.
 - Specify the **Type** as **String** and set a default value as any text between quotes, followed by a semicolon, e.g. `"This is my text";`

- **Extraction:**
 - In the **Data Model**, select a field.
 - On the **Step properties** pane, under **Field Definition** set the **Type** to **String**. The field value will be extracted and treated as a string.
- **JavaScript Expression:** Set the desired value to any string between quotes. Example:


```
record.fields["countryOfOrigin"] = "Canada";
```

Building String values

String values can be made up of more than just a series of characters between quotes. Here are a few tips and tricks to build strings:

- Both single and double quotes can be used to surround strings and they will act in precisely the same manner. So, "this is a string" and 'this is a string' mean the same thing. However, it's useful to have both in order to remove the need for escaping characters. For instance, "I'm fine!" works, but 'I'm fine!' does not since only 'I' is properly interpreted. '\I'm fine!' works (escaping the ' with a \).
- It is possible to put more than one string, as well as variables containing strings, by concatenating them with the + operator. For example, "Hello " + sourceRecord.property.FirstName + ", nice to meet you!".
- Adding more data to an existing string variable or field is possible using a combination of concatenation and assignment. For example, after the statements `var myVar = "Is this the real life";`, and `myVar += " or is this just fantasy?";`, the value of `myVar` will be, obviously "Is this the real life or is this just fantasy?".

For more information on string variables, see quicksmode.org.

Data Model file structure

The Data Model file is an XML file that contains the structure of the Data M model, including each field's name, data type, and any number of detail tables and nested tables.

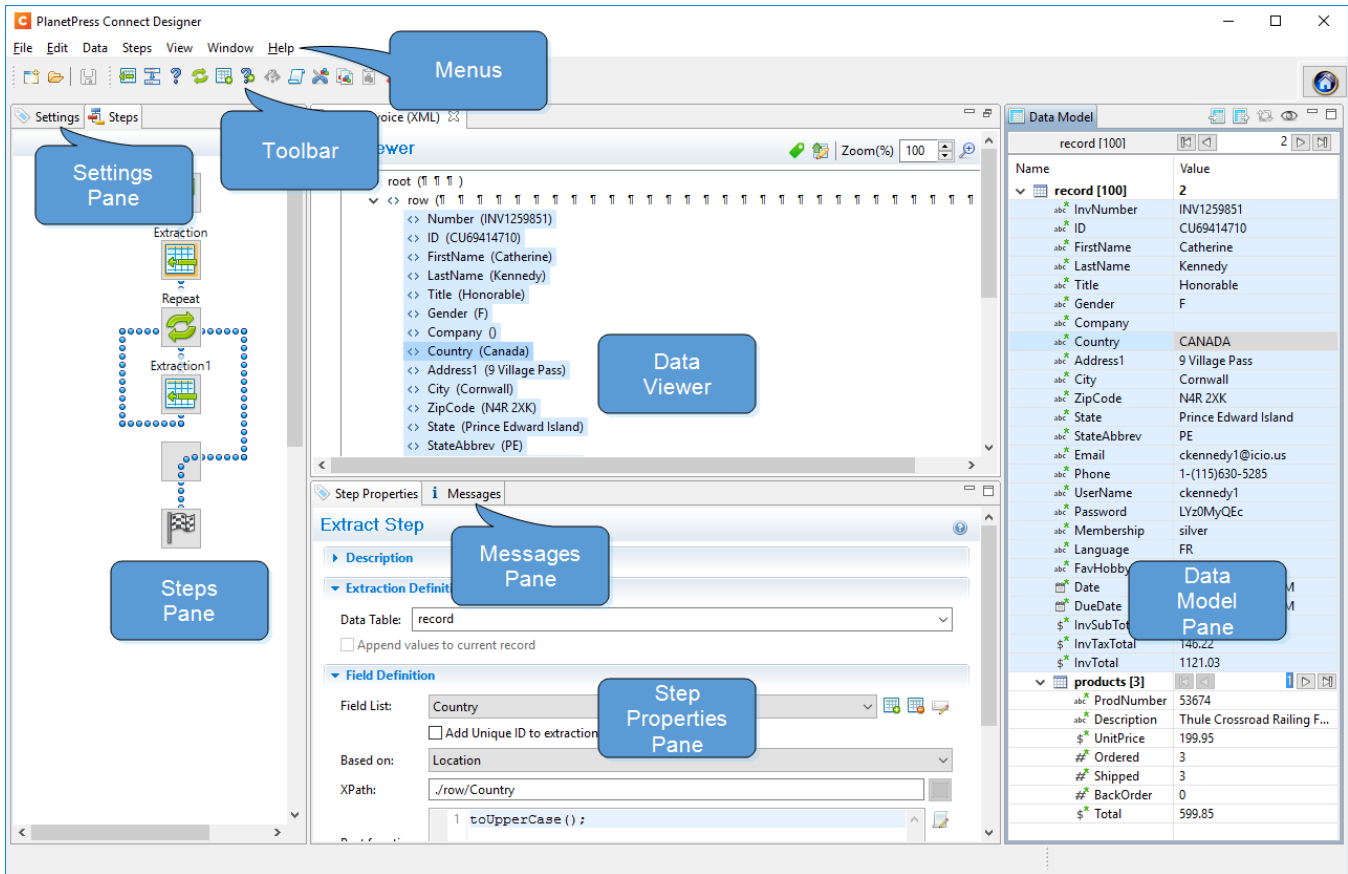
Example: promotional data

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<datamodel schemaVersion="1.0.0.3" name="Generic Address Block"
version="1"
xmlns="http://www.objectiflune.com/connectschemas/DataModelConfig"
```

```
xsi:schemaLocation="http://www.objectiflune.com/connectschemas/Data
ModelConfig
http://www.objectiflune.com/connectschemas/DataModelConfig/1_0_0_
3.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <field type="string" name="Name" required="true"/>
  <field type="string" name="Organization" required="true"/>
  <field type="string" name="Address1" required="true"/>
  <field type="string" name="Address2" required="true"/>
  <field type="string" name="Address3" required="true"/>
  <field type="string" name="City" required="true"/>
  <field type="string" name="StateOrProvince" required="true"/>
  <field type="string" name="Country" required="true"/>
  <field type="string" name="ZipOrPostalCode" required="true"/>
  <field type="string" name="Extra1" required="true"/>
  <field type="string" name="Extra2" required="true"/>
</datamodel>
```

Example: transactional details, in a simple invoice format

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<datamodel schemaVersion="1.0.0.3" name="Transactional Invoice"
version="1"
xmlns="http://www.objectiflune.com/connectschemas/DataModelConfig"
xsi:schemaLocation="http://www.objectiflune.com/connectschemas/Data
ModelConfig
http://www.objectiflune.com/connectschemas/DataModelConfig/1_0_0_
3.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <field type="string" name="ID" required="true"/>
  <field type="string" name="Gender" required="true"/>
  <field type="string" name="LastName" required="true"/>
  <field type="string" name="FirstName" required="true"/>
  <field type="string" name="Address1" required="true"/>
  <field type="string" name="Address2" required="true"/>
  <field type="string" name="City" required="true"/>
  <field type="string" name="State" required="true"/>
  <field type="string" name="Country" required="true"/>
  <field type="string" name="ZipCode" required="true"/>
  <field type="string" name="Title" required="true"/>
  <field type="string" name="Company" required="true"/>
  <field type="string" name="Phone2" required="true"/>
  <field type="string" name="Email" required="true"/>
  <field type="string" name="Language" required="true"/>
  <field type="string" name="Number" required="true"/>
  <field type="datetime" name="Date" required="true"/>
  <field type="datetime" name="DueDate" required="true"/>
  <field type="string" name="Rep" required="true"/>
  <field type="currency" name="SubTotal" required="true"/>
  <field type="currency" name="TaxTotal" required="true"/>
  <field type="currency" name="Total" required="true"/>
  <table name="detail">
    <field type="string" name="Number2" required="true"/>
    <field type="string" name="Description" required="true"/>
    <field type="currency" name="UnitPrice" required="true"/>
    <field type="integer" name="Ordered" required="true"/>
    <field type="integer" name="Shipped" required="true"/>
    <field type="integer" name="BackOrder" required="true"/>
    <field type="currency" name="Total2" required="true"/>
  </table>
</datamodel>
```

Keyboard shortcuts

This topic gives an overview of keyboard shortcuts that can be used in the DataMapper. Keyboard shortcuts available in the Designer for menu items, script editors and the data model pane can also be used in the DataMapper; see "Keyboard shortcuts" on page 738. Although some of the keyboard shortcuts are the same, this isn't a complete list of Windows keyboard shortcuts. Please refer to Windows documentation for a complete list of Windows keyboard shortcuts.

Menu items

The following key combinations activate a function in the menu.

Key combination	Function
Alt	Put the focus on the menu. (Alt + the underlined letter in a menu name)

Key combination	Function
	displays the corresponding menu.) The menu can then be browsed using the Enter key, arrow up and arrow down buttons.
Alt + F4	Exit
Ctrl + C or Ctrl + Insert	Copy
Ctrl + N	New
Ctrl + O	Open file
Ctrl + Shift + O	Open configuration file
Ctrl + S	Save file
Ctrl + V or Shift + Insert	Paste
Ctrl + X	Cut
Ctrl + W or Ctrl + F4	Close file
Ctrl + Y or Ctrl + Shift + Y	Redo
Ctrl + Z or Ctrl + Shift + Z	Undo

Key combination	Function
Ctrl + Shift + S	Save all
Ctrl + Shift + W or Ctrl + Shift + F4	Close all
Ctrl + F5	Revert
Ctrl + F7	Next view
Ctrl + Shift + F7	Previous view
Ctrl + F8	Next perspective
Ctrl + Shift + F8	Previous perspective
Ctrl + F10	Save as
Ctrl + F12	Send to Workflow / Package files
F4	Ignore step
F6	Add an Extract step
F7	Add a Goto step
F8	Add a Condition step
F9	Add a Repeat step

Key combination	Function
F10	Add an Extract field
F11	Add an Action step
F12	Add a Multiple Conditions step
Alt + F12	Add a Case step (under a Multiple Conditions step)
Home	Go to the first step in the workflow
End	Go to the last step in the workflow
Alt + V	Validate records
Shift + F10 or Ctrl + Shift + F10	Open context menu

Viewer pane

The following key combinations activate a function in the **Viewer**.

Key combination	Function
Alt + -	Open system menu
Ctrl + -	Zoom out
Ctrl + +	Zoom in
Ctrl + Shift + E	Switch to Editor
Ctrl + F6	Next editor (when there is more than one file open in the Workspace)

Key combination	Function
Ctrl + Shift + F6	Previous editor (when there is more than one file open in the Workspace)

Data Model pane

Key combination	Function
PageUp	Go to previous record
PageDown	Go to next record
Alt + CR	Property page
Alt + PageDown	Scroll down to the last field
Alt + PageUp	Scroll up to the first field

Steps tab

Key combination	Function
Ctrl + -	Zoom out
Ctrl + +	Zoom in

Edit Script and Expression windows

The following key combinations have a special function in the **Expression** and in the **Edit Script** windows (expanded view).

Key combination	Function
Ctrl + space	Content assist (auto-complete)

Key combination	Function
Ctrl + A	Select all
Ctrl + D	Duplicate line
Ctrl + I	Indent (Tab)
Ctrl + J	Line break
Ctrl + L	Go to line; a prompt opens to enter a line number.
Ctrl + Shift + D	Delete line
Shift + Tab	Shift selected lines left
Tab	Shift selected lines right
Ctrl + /	Comment out / uncomment a line in code
Ctrl + Shift + /	Comment out / uncomment a code block

Menus

The following menu items are shown in the DataMapper Module's menu:

File Menu

- **New...:** Opens the [Creating a New Data Mapping Configuration](#) dialog.
- **Open:** Opens a standard File Open dialog. This dialog can be used to open Templates and data mapping configurations.
- **Open Recent:** List the most recently opened Templates and configurations. Clicking on a template will open it in the Designer module, clicking on a data mapping configuration will open it in the DataMapper module.

- **Close:** Close the currently open data mapping configuration or Template. If the file needs to be saved, the appropriate Save dialog will open.
- **Close All:** Close any open data mapping configuration or Template. If any of the files need to be saved, the Save Resources dialog opens.
- **Save:** Saves the current data mapping configuration or Template to its current location on disk. If the file is a data mapping configuration and has never been saved, the Save As dialog appears instead.
- **Save As...:** Saves the current data mapping configuration or Template to a new location on disk. In the case of Templates, it is saved to a location that can be different than the local repository.
- **Save All:** Saves all open files. If any of the open files have never been saved, the Save As dialog opens for each new unsaved file.
- **Revert:** Appears only in the Designer module. Reverts all changes to the state in which the file was opened or created.
- **Add Data:** Adds data either to the current data mapping configuration or to the open template. In data mapping configuration
 - **From File...:** Opens the dialog to add a new data file to the currently loaded data mapping configuration. Not available if the currently loaded data mapping configuration connects to a database source.
 - **From Database...:** Opens the Edit Database Configuration dialog. Not available if the currently loaded data mapping configuration is file-based.
- **Send to Workflow:** Opens the [Send to Workflow](#) dialog to send files to a local PlanetPress Workflow software installation.
- **Exit:** Closes the software. If any of the files need to be saved, the Save Resources dialog opens.

Edit Menu

- **Undo:** Undoes the previous action.
- **Redo:** Redoes the last action that was undone.
- **Cut Step:** Removes the currently selected step and places it in the clipboard. If the step is a Repeat or a Condition, all steps under it are also placed in the clipboard. If there is already a step in the clipboard, it will be overwritten.
- **Copy Step:** Places a copy of the currently selected step in the clipboard. The same details as the Cut step applies.

- **Paste Step:** Takes the step or steps in the clipboard and places them in the Steps after the currently selected step.
- **Delete Step:** Deletes the currently selected step. If the step is a Repeat or Condition, all steps under it are also deleted.
- **Cut:** Click to remove the currently selected step, or steps, and place them in the clipboard.
- **Copy:** Click to place a copy of the currently selected step, or steps, in the clipboard.
- **Paste:** Click to place any step, or steps, from the clipboard before the currently selected step in the [Steps Pane](#).

Data Menu

- **Hide/Show datamap:** Click to show or hide the icons to the left of the Data Viewer that displays how the steps affect the line.
- **Hide/Show extracted data:** Click to show or hide the extraction selections indicating that data is extracted. This simplifies making data selections in the same areas and is useful to display the original data.
- **Validate All Records:** Runs the Steps on all records and verifies that no errors are present in any of the records. Errors are displayed in the [Messages Pane](#).

Steps

- **Ignore Step:** Click to set the step to be ignored (aka disabled). Disabled steps do not run when in DataMapper and do not execute when the data mapping configuration is executed in Workflow. However, they can still be modified normally.
- **Add Extract Step:** Adds an Extract Step with one or more extract fields. If more than one line or field is selected in the Data Viewer, each line or field will have an extract field.
- **Add Goto Step:** Adds a Goto step that moves the selection pointer to the beginning of the data selection. For instance if an XML node is selected, the pointer moves to where that node is located.
- **Add Condition Step:** Adds a condition based on the current data selection. The "True" branch gets run when the text is found on the page. Other conditions are available in the step properties once it has been added.
- **Add Repeat Step:** Adds a loop that is based on the current data selection, and depending on the type of data. XML data will loop on the currently selected node, CSV loops for all rows in the record. In Text and PDF data, if the data selection is on the same line as the cursor position, the loop will be for each line until the end of the record. If the

data selection is on a lower line, the loop will be for each line until the text in the data selection is found at the specified position on the line (e.g. until "TOTAL" is found).

- **Add Extract Field:** Adds the data selection to the selected Extract step, if an extract step is currently selected. If multiple lines, nodes or fields are selected, multiple extract fields are added simultaneously.
- **Add Multiple Conditions:** Adds a condition that splits into multiple case conditions.
- **Add Action Step:** Adds a step to run one or more specific actions such as running a JavaScript expression or setting the value of a Source Record Property.

View Menu

- **Zoom In:** Click to zoom in the [Steps Pane](#).
- **Zoom Out:** Click to zoom out the [Steps Pane](#).

Window Menu

- **Show View**
 - **Messages:** Shows the [Messages Pane](#).
 - **Steps:** Shows the [Steps Pane](#).
 - **Settings:** Shows the [Settings Pane](#).
 - **Record:** Shows the Record Pane.
 - **Detail tables :** Each detail table and nested table is listed here. Click on one to show it in the [Data Model Pane](#).
 - **Step Properties:** Shows the [Step Properties Pane](#).
- **Reset Perspective:** Resets all toolbars and panes to the initial configuration of the module.
- **Preferences:** Click to open the [Preferences](#) dialog.

Help Menu

- **Software Activation:** Displays the Software Activation dialog. See [Activating your license](#).
- **Help Topics:** Click to open this documentation.
- **Contact Support:** Click to open the [Objectif Lune Contact Page](#) in the default system Web browser.

- **About PlanetPress Connect Designer:** Displays the software's About dialog.
- **Welcome Screen:** Click to re-open the Welcome Screen.

Panes

The DataMapper screen contains the following panes.

- "Settings pane" on page 203. The Settings pane contains settings for the data source.
- "Steps pane" on page 213. The entire extraction workflow is visible in the Steps pane.
- "The Data Viewer" on page 200. The Data Viewer shows one record in the data source.
- "Step properties pane" on page 215. The Step properties pane contains all settings for the step that is currently selected on the Steps pane.
- "Data Model pane" below. The Data Model pane shows one extracted record.
- "Messages pane" on page 202.

Data Model pane

The Data Model pane displays the result of all the preparations and extractions of the extraction workflow. The pane displays the content of a single record within the record set at a time.

Data is displayed as a tree view, with the root level being the record table. On the level below that are detail tables, and a detail table inside a detail table is called a nested table.

The Data Model is also used as a navigation tool between records and in all detail tables.

Data Model toolbar buttons



: **Import** a Data Model.



: **Export** the Data Model.



: **Synchronize** the Data Model and the data sample.




: Show the **ExtraData** field. Note that this field is not meant to be filled via an extraction. It can be used in Workflow to add data to the Data Model; see "About adding fields and data via Workflow" on page 154.

Data Model contextual menu

The Data Model is generally constructed by extracting data; see "Extracting data" on page 118. It is however possible to modify the Data Model, even if no data is present in the pane. To do

this, open the contextual menu within the pane itself by right-clicking on something in the Data Model pane. Depending on where you've clicked, it can contain the following options:

- **Add a field:** Click to add a new field at the current level (record or detail table). Enter the field name in the dialog and click OK to add it.
- **Add a table:** Click to add a new detail table at the current level (record or existing detail table). Enter the table name in the dialog and click OK to add it.
- **Default Value:** Click to set the default value for a field. This value is used if no extraction is present, or if an extraction attached to this field returns no value.
- **Collapse Fields:** Collapse the fields in the selected level.
- **Expand Fields:** Clicking the icon that represents collapsed fields (for example:  [19]) enables this menu item. It is used to expand the fields on one level.
- **Collapse All Fields:** Collapse the fields on the record level and in all detail tables.
- **Expand All Fields:** Expand the fields on the record level and in all detail tables.

Note

The following options are only available for Data Model fields or detail tables that are **not** filled via an extraction.

Fields and detail tables that are filled via an Extract step are to be changed (renamed, deleted etc.) via the properties of that Extract step; see: "Editing fields" on page 158 and "Renaming a detail table" on page 193.

- **Rename:** Click to rename the selected table or field. Enter the new name and click OK to rename.
- **Delete:** Click to delete the selected table or field.
- **Set Type:** Use the list to select the field type (see "Data types" on page 168).

Field display

Fields in the Data Model pane are displayed in specific ways to simplify comprehension of the display data:

- **Value:** The current value of the extracted field, based on the record shown in the Data Viewer.

- The column on the left displays the **name** of the field.
- The column on the right displays the current **value** of the extracted field based on the record shown in the Data Viewer, if an Extract step has an extraction for this field (see "Extracting data" on page 118).
- The **icon** to the left of the name indicates the **data Type** of the field (see "Data types" on page 168).
- A field name with an **asterisk** to the right indicates that this field is part of an imported Data Model file.
- A field with a **grey background** indicates this Data Model field does not have any attached extracted data.
- A field with a **white background** indicates that the field has attached extracted data but the step extracting the data is not currently selected.
- A field with a **blue background** indicates that the field has attached extracted data and the step extracting the data is currently selected.

Record navigation

Records can be navigated via the Data Model pane. The default record level navigates between records both in the Data Model pane and the Data Viewer, while each detail table has a similar navigation that influences that table and each detail table under it.

- **Expand/Contract:** Click to hide or show any fields or tables under the current table level.
- **Table Name:** Displays the name of the table as well as the number of records at that level (in [brackets]). At the record level this is the number of records. In other levels it represents the number of entries in a detail table.
- **Number of Records:** The number of available records in the active data sample. This is affected by the Boundary settings (see "Record boundaries" on page 117 and "Settings pane" on page 203) and the Preprocessor step ("Preprocessor step" on page 140).
- **First Record:** Go to the first record in the data sample. This button is disabled if the first record is already shown.
- **Previous Record:** Go to the previous record in the data sample. This button is disabled if the first record is shown.
- **Current Record:** Displays the current record or table entry. Type a record number and press the **Enter** key to display that record. The number has to be within the number of available records in the data sample.

- **Next Record:** Go to the next record in the data sample. This button is disabled if the last record is shown.
- **Last Record:** Go to the last record in the data sample. This button is disabled if the last record is already shown. If a record limit is set in the Settings pane ("Settings pane" on page 203) the last record will be within that limit.

Detail tables

A detail table is a field in the Data Model that contains a record set instead of a single value. Detail tables contain transactional data. They are created when an **Extract** step is added within a **Repeat** step; see "Extracting transactional data" on page 124.

In the most basic of transactional communications, a single detail table is sufficient. However, it is possible to create multiple detail tables, as well as nested tables. Detail tables and nested tables are displayed as separate levels in the Data Model (see "The Data Model" on page 151).

Renaming a detail table

Renaming detail tables is especially useful when there are more detail tables in one record, or when a detail table contains another detail table. For this detail table, 'products' would be a better name.

1. On the Data Model pane, click one of the fields in the detail table.
2. On the Step Properties pane, under **Extraction Definition**, in the **Data Table** field, you can find the name of the detail table: **record.detail** by default. Change the **detail** part in that name into something else.

Note

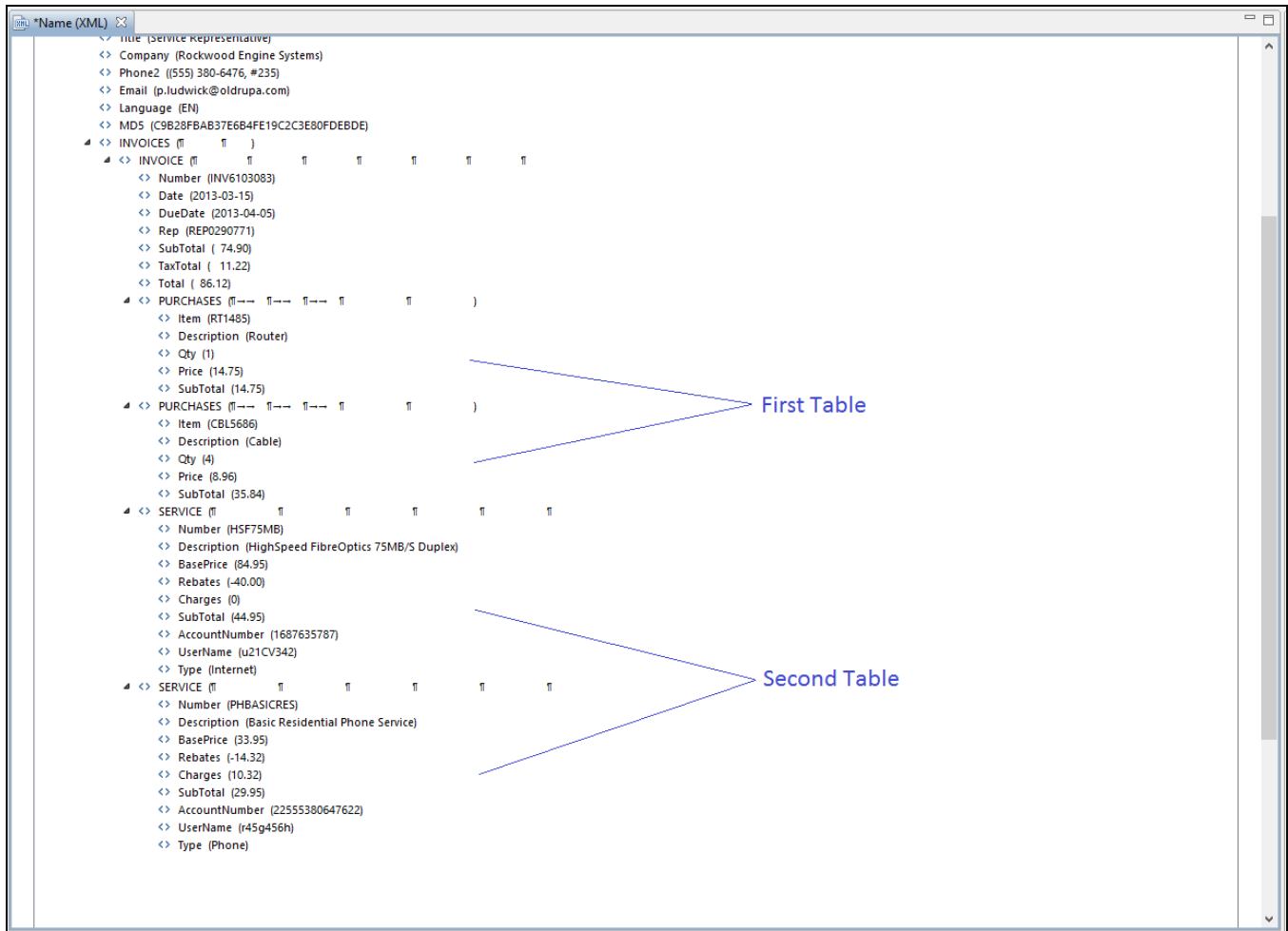
A detail table's name should always begin with '**record.**'

3. Click somewhere else on the Step Properties pane to update the Data Model. You will see the new name appear.

Creating multiple detail tables

Multiple detail tables are useful when more than one type of transactional data is present in the source data, for example purchases (items with a set price, quantity, item number) and services

(with a price, frequency, contract end date, etc).



To create more than one detail table, simply extract transactional data in different Repeat steps (see "Extracting transactional data" on page 124).

The best way to do this is to add an empty detail table (right-click the Data Model, select **Add a table** and give the detail table a name) and drop the data on the name of that detail table. Else the extracted fields will all be added to one new detail table with a default name at first, and you will have to rename the detail table created in each Extract step to pull the detail tables apart (see "Renaming a detail table" on the previous page).

Steps

- Extraction
- Extraction1

***Name (XML)**

```

<> Phone2 ((555) 702-5520, #665)
<> Email (a.tincher@oldrupa.com)
<> Language (EN)
<> MD5 (BE9479B978EE0AE3970837291B7105C8)
<> INVOICES (⌈ ⌋)
  <> INVOICE (⌈ ⌋)
    <> Number (INV8559825)
    <> Date (2013-03-15)
    <> DueDate (2013-04-05)
    <> Rep (REP8800122)
    <> SubTotal ( 125.96)
    <> TaxTotal ( 18.86)
    <> Total ( 144.82)
  <> PURCHASES (⌈--- ⌋--- ⌋--- ⌋)
    <> Item (RT1485)
    <> Description (Router)
    <> Qty (1)
    <> Price (14.75)
    <> SubTotal (14.75)
  <> PURCHASES (⌈--- ⌋--- ⌋--- ⌋)
    <> Item (CBL5686)
    <> Description (Cable)
    <> Qty (3)
    <> Price (8.96)
    <> SubTotal (26.88)
  <> PURCHASES (⌈--- ⌋--- ⌋--- ⌋)
    <> Item (FST5678)
    <> Description (Fasteners)
    <> Qty (50)
    <> Price (0.25)
    <> SubTotal (12.5)
  <> <SRVICF (⌈ ⌋)
  
```

Data model

Name	Value
record [3]	2
abc FULLNAME	Amie Tincher
abc ID	CU44643835
abc Gender	Miss
abc LastName	Tincher
abc FirstName	Amie
abc Address1	356, Chambers Street
abc Address2	P.O. Box 6049
abc City	St. Mary's
abc State	ON
abc Country	CA
abc ZipCode	L6J 5L1
abc Title	Driver/Sales Workers
abc Company	St. Mary's Pest Control Ltd
abc Phone2	(555) 702-5520, #665
abc Email	a.tincher@oldrupa.com
abc Language	EN
abc MD5	BE9479B978EE0AE397083...
abc Number	INV8559825
abc Date	2013-03-15
abc DueDate	2013-04-05
abc Rep	REP8800122
abc SubTotal	125.96
abc TaxTotal	18.86
abc Total	144.82

Step Properties

Extract Step

Description

Extraction Definition

Data Table: record

Append values to current record

Steps

- Extraction
- Extraction1
- Repeat
- Extraction2
- Repeat1
- Extraction3

***Name (XML)**

```

<> SubTotal (14.75)
<> PURCHASES (⌈--- ⌋--- ⌋--- ⌋)
  <> Item (CBL5686)
  <> Description (Cable)
  <> Qty (3)
  <> Price (8.96)
  <> SubTotal (26.88)
<> PURCHASES (⌈--- ⌋--- ⌋--- ⌋)
  <> Item (FST5678)
  <> Description (Fasteners)
  <> Qty (50)
  <> Price (0.25)
  <> SubTotal (12.5)
<> SERVICE (⌈ ⌋)
  <> Number (HSF75MB)
  <> Description (HighSpeed FibreOptics 75MB/S Duplex)
  <> BasePrice (84.95)
  <> Rebates (-40.00)
  <> Charges (0)
  <> SubTotal (44.95)
  <> AccountNumber (1687635787)
  <> UserName (u21CV342)
  <> Type (Internet)
<> SERVICE (⌈ ⌋)
  <> Number (PHBASICSRES)
  <> Description (Basic Residential Phone Service)
  <> BasePrice (33.95)
  <> Rebates (-14.32)
  <> Charges (22.32)
  <> SubTotal (41.95)
  <> AccountNumber (22555705552022)
  <> UserName (w457478)
  
```

Data model

Name	Value
record [3]	2
abc FULLNAME	Amie Tincher
abc ID	CU44643835
abc Gender	Miss
abc LastName	Tincher
abc FirstName	Amie
abc Address1	356, Chambers Street
abc Address2	P.O. Box 6049
abc City	St. Mary's
abc State	ON
abc Country	CA
abc ZipCode	L6J 5L1
abc Title	Driver/Sales Workers
abc Company	St. Mary's Pest Control Ltd
abc Phone2	(555) 702-5520, #665
abc Email	a.tincher@oldrupa.com
abc Language	EN
abc MD5	BE9479B978EE0AE397083...
abc Number	INV8559825
abc Date	2013-03-15
abc DueDate	2013-04-05
abc Rep	REP8800122
abc SubTotal	125.96
abc TaxTotal	18.86
abc Total	144.82
detail [6]	1
abc Item	RT1485
abc Description	Router
abc Qty	1
abc Price	14.75
abc SubTotal2	14.75
abc Number2	
abc Description2	
abc BasePrice	
abc Rebates	
abc Charges	
abc SubTotal3	
abc AccountNum	
abc UserName	
abc Type	

Step Properties

Extract Step

Description

Extraction Definition

Data Table: record.detail

Append values to current record

Nested detail tables

Nested detail tables are used to extract transactional data that are relative to other data. They are created just like multiple detail tables, with two differences:

- For the tables to be actually nested, the **Repeat** step and its **Extract** step that extract the nested transactional data must be located **within** the **Repeat** step that extracts data to a detail table.
- In their name, the dot notation (record.services) must contain one extra level (record.services.charges).

Note

Using nested detail tables in the **Designer** module requires scripting, as described in this How-to: [Cloning your way through nested tables](#).

Example

An XML source file lists the services of a multi-service provider: Internet, Cable, Home Phone, Mobile. Each service in turn lists a number of "charges", being service prices and rebates, and

a number of "details" such as movie rentals or long distance calls.

XML viewer

- <> Rep (REP8800122)
- <> SubTotal (93.76)
- <> TaxTotal (14.04)
- <> Total (107.80)
- ▲ <> SERVICE (¶ ¶ ¶ ¶)
 - <> Number (TVDIGHD)
 - <> Description (High Definition Digital TV)
 - <> BasePrice (30.00)
 - <> Rebates (-28.19)
 - <> Charges (67.25)
 - <> ExtraServices (24.70)
 - <> AccountNumber (6784589574527852)
 - <> SubTotal (39.06)
 - <> Type (Television)
 - ▲ <> Charges (¶ ¶ ¶)
 - ▲ <> CHARGE (¶ ¶ ¶ ¶)
 - <> ID (RBTVCRTCFAPL)
 - <> Description (Contribution to CRTC FAPL)
 - <> Price (-0.13)
 - ▲ <> CHARGE (¶ ¶ ¶ ¶)
 - <> ID (RBTVPROG)
 - <> Description (PROMO: Programming Credit)
 - <> Price (-14.20)
 - ▲ <> CHARGE (¶ ¶ ¶ ¶)
 - <> ID (RBTVHDBOX)
 - <> Description (PROMO: Free HD Receiver Rental)
 - <> Price (-13.86)
 - ▲ <> DETAILS (¶ ¶ ¶ ¶)
 - ▲ <> DETAIL (¶ ¶ ¶ ¶)
 - <> CODE (MVHDLOC)
 - <> ID (55123)
 - <> Description (Gangster Squad HD)
 - <> TimeStamp (2013-02-16 19:24:33)
 - <> Price (5.95)
 - ▲ <> DETAIL (¶ ¶ ¶ ¶)
 - <> CODE (MVHDLOC)
 - <> ID (55234)
 - <> Description (Indiana Jones: Raiders of the Lost Ark)
 - <> TimeStamp (2013-02-23 18:02:46)
 - <> Price (2.95)
 - ▲ <> DETAIL (¶ ¶ ¶ ¶)
 - <> CODE (MVHDLOC)
 - <> ID (55345)
 - <> Description (Indiana Jones and the Temple of Doom)

Data to be Nested

The services can be extracted to a detail table called **record.services**.

The screenshot displays an XML viewer on the left and a data model on the right. The XML viewer shows a tree structure for an invoice, including customer details, invoice summary, and service details. The data model on the right shows a table with columns for Name and Value, containing fields like FULLNAME, ID, Gender, Address, and Invoice totals.

Name	Value
record [3]	3
FULLNAME	Melba Lumpkin
ID	CU35692808
Gender	Miss
LastName	Lumpkin
FirstName	Melba
Address1	564, Hidden Valley Drive
Address2	Suite 9530
City	Gore Bay
State	ON
Country	CA
ZipCode	F3U 5B3
Title	Machine Tool Cutting Oper...
Company	Gore Bay Game Group
Phone2	(555) 631-670, #854
Email	m.lumpkin@oldrupa.com
Language	EN
MD5	A376734A49841E9F04696F8E...
Number	INV4839392
Date	2013-03-15
DueDate	2013-04-05
Rep	REP8800122
SubTotal	93.76
TaxTotal	14.04
Total	107.80

The "charges" and "details" can be extracted to two nested detail tables.

The screenshot shows a workflow editor on the left with steps: Extraction, Extraction1, Repeat, Extraction2, and a final data output step. The central XML viewer displays the following structure:

```

< INVOICE (INV4839392)
  < Number (INV4839392)
  < Date (2013-03-15)
  < DueDate (2013-04-05)
  < Rep (REP8800122)
  < SubTotal ( 93.76)
  < TaxTotal ( 14.04)
  < Total ( 107.80)
  < SERVICE (TVDIGHD)
    < Number (TVDIGHD)
    < Description (High Definition Digital TV)
    < BasePrice (30.00)
    < Rebates (-28.19)
    < Charges (67.25)
    < ExtraServices (24.70)
    < AccountNumber (6784589574527852)
    < SubTotal (39.06)
    < Type (Television)
    < Charges ( )
      < CHARGE ( )
        < ID (RBTVCRTCFAPL)
        < Description (Contribution to CRTFC FAPL)
        < Price (-0.13)
      < CHARGE ( )
        < ID (RBTVPROG)
        < Description (PROMO: Programming Credit)
        < Price (-14.20)
      < CHARGE ( )
        < ID (RBTVHDBOX)
        < Description (PROMO: Free HD Receiver Rental)
        < Price (-13.86)
    < DETAILS ( )
      < DETAIL ( )
        < CODE (MVHDLOC)
        < ID (55123)
        < Description (Gangster Squad HD)
        < Timestamp (2013-02-16 19:24:33)
        < Price (5.95)
      < DETAIL ( )
        < CODE (MVHDLOC)
        < ID (55234)
        < Description (Indiana Jones: Raiders of the Lost Ark)
        < Price (7.95)
  
```

The data model on the right shows a table with columns 'Name' and 'Value'. It contains a 'record [3]' section with fields like FULLNAME, ID, Gender, etc., and a 'detail [1]' section with fields like Number2, Description, BasePrice, etc.

The nested tables can be called **record.services.charges** and **record.services.details**.

This screenshot shows a more complex workflow with steps: Extraction, Extraction1, Repeat, Extraction2, Repeat1, Extraction3, Repeat2, Extraction4, and a final data output step. The XML viewer displays a similar structure to the first screenshot, but with a different 'DETAIL' entry:

```

< SERVICE (TVDIGHD)
  < Number (TVDIGHD)
  < Description (High Definition Digital TV)
  < BasePrice (30.00)
  < Rebates (-28.19)
  < Charges (67.25)
  < ExtraServices (24.70)
  < AccountNumber (6784589574527852)
  < SubTotal (39.06)
  < Type (Television)
  < Charges ( )
    < CHARGE ( )
      < ID (RBTVCRTCFAPL)
      < Description (Contribution to CRTFC FAPL)
      < Price (-0.13)
    < CHARGE ( )
      < ID (RBTVPROG)
      < Description (PROMO: Programming Credit)
      < Price (-14.20)
    < CHARGE ( )
      < ID (RBTVHDBOX)
      < Description (PROMO: Free HD Receiver Rental)
      < Price (-13.86)
  < DETAILS ( )
    < DETAIL ( )
      < CODE (MVHDLOC)
  
```

The data model on the right shows a table with columns 'Name' and 'Value'. It contains a 'record [3]' section and a 'details [10]' section with fields like Number2, Description, BasePrice, etc.

The Preprocessor window at the bottom shows a table for 'Fixed automation properties' and a 'Preprocessor' table:

Name	Scope	Type	Default Value

Name	Type

Now one "charges" table and one "details" table are created for each row in the "services" table.

The Data Viewer

The Data Viewer is located in the middle on the upper half of the DataMapper screen. It displays the data source that is currently loaded in the DataMapper, specifically one record in that data. Where one record ends and the next starts, is set in the Data Source settings (see "Record boundaries" on page 117). One record may contain more than one unit: PDF or Text pages, XML nodes, CSV lines, etc.

When the Delimiter or Boundary options are set in the Settings pane, the Data Viewer reflects those changes.

Any modification of the source data by a Preprocessor takes place before the data is displayed in the Data Viewer (see "Preprocessor step" on page 140).

The Data Viewer lets you select data, extract them ("Extracting data" on page 118), and apply a condition where necessary.

How data can be selected depends on the type of source file (see "Selecting data" on page 122).

Once data is extracted, clicking on any Extract step on the **Steps** pane highlights any area from which it extracts data in the Data Viewer. You can click on the Preprocessor step to select all the steps in the extraction workflow and highlight all extracted data.

Clicking on other step types also has a visible effect in the Data Viewer:

- Clicking on a Repeat step shows where the loop takes place.
- Clicking on a Goto step shows where the cursor is moved.
- Clicking on a Condition step shows which data fulfil the condition.

For more information about the different steps that can be added to a data mapping workflow, see "Steps" on page 140.

Data Viewer toolbar

The Data Viewer has a **toolbar** at the top to control options in the viewer. Which toolbar features are available depends on the data source type.

- **Font** (Text file only): Use the drop-down to change the font used to display text. Useful for double-byte data. It is recommended that monospace fonts be used.

- **Hide/Show line numbers #** (Text file only): Click to show or hide the line numbers on the left of the Data Viewer.
- **Hide/Show datamap** 📍: Click to show or hide the icons to the left of the Data Viewer which displays how the steps affect the line.
- **Hide/Show extracted data** 📄: Click to show or hide the extraction selections indicating that data is extracted. This simplifies making data selections in the same areas and is useful to display the original data.
- **Lock/Unlock extracted data** 🔒: Click to lock existing extraction selections so they cannot be moved or resized. This simplifies making data selections in the same area.
- **Zoom Level**: Use the arrows to adjust the zoom level, or type in the zoom percentage.
- **Zoom In (CTRL +)** 🔍+: Click to zoom in by increments of 10%
- **Zoom Out (CTRL -)** 🔍-: Click to zoom out by increments of 10%

Additional Keyboard Shortcuts for XML Files:

- **+** (while on an XML node with children): Expand the XML Node
- **-** (while on an XML node with children): Collapse the XML node, hiding all its children nodes.

Contextual Menu

You can access the contextual menu using a right-click anywhere inside the Viewer window.

Add Extraction	F6
Add Goto	F7
Add Conditional	F8
Add Repeat	F9
Add Action	F11
Add Extract Field	F10

Note

The **Add Extract Field** item is available only after an **Extract** step has been added to the workflow.

Messages pane

The Messages pane is shared between the DataMapper and Designer modules and displays any warnings and errors from the data mapping configuration or template.

At the top of the Message pane are control buttons:

- **Export Log:** Click to open a Save As dialog where the log file (.log) can be saved on disk.
- **Clear Log Viewer:** Click to remove all entries in the log viewer.
- **Filters:** Displays the Log filter (see "Log filter" below).
- **Activate on new events:** Click to disable or enable the automatic display of this dialog when a new event is added to the pane.
- **Time:** The date and time when the error occurred.
- **Type:** Whether the entry is a warning or an error.
- **Source:** The source of the error. This indicates the name of the step as defined in its step properties.
- **Message:** The contents of the message, indicating the actual error.

Log filter

The log filter determines what kind of events are show in the Messages pane (see "Messages pane" above).

- **Event Types** group:
 - **OK:** Uncheck to hide OK-level entries.
 - **Information:** Uncheck to hide information-level entries.
 - **Warning:** Uncheck to hide any warnings.
 - **Error:** Uncheck to hide any critical errors.
- **Limit visible events to:** Enter the maximum number of events to show in the Messages Pane. Default is 50.

Settings pane

Settings for the data source and a list of Data Samples and JavaScript files used in the current data mapping configuration, can be found on the Settings tab at the left. The available options depend on the type of data sample that is loaded.

The Input Data settings (especially Delimiters) and Boundaries are essential to obtain the data and eventually, the output that you need. For more explanation, see "Data source settings" on page 115.

Input Data

The **Input Data** settings specify how the input data must be interpreted. These settings are different for each data type. For a CSV file, for example, it is important to specify the delimiter that separates data fields. PDF files are already delimited naturally by pages, so the input data settings for PDF files are interpretation settings for text in the file.

CSV file Input Data settings

In a CSV file, data is read line by line, where each line can contain multiple fields. The input data settings specify to the DataMapper module how the fields are separated.

- **Field separator:** Defines what character separates each field in the file. Even though CSV stands for comma-separated values, CSV can actually refer to files where fields are separated using any character, including commas, tabs, semicolons, and pipes.
- **Text delimiter:** Defines what character surrounds text in the file, preventing the **Field separator** from being interpreted within those text delimiters. This ensures that, for example, the field "Smith; John" is not interpreted as two fields, even if the field delimiter is the semicolon.
- **Comment delimiter:** Defines what character starts a comment line.
- **Encoding:** Defines what encoding is used to read the Data Source (US-ASCII, ISO-8859-1, UTF-8, UTF-16, UTF-16BE or UTF-16LE).
- **Lines to skip:** Defines a number of lines in the CSV that will be skipped and not used as records.
- **Set tabs as a field separator:** Overwrites the **Field separator** option and sets the Tab character instead for tab-delimited files.
- **First row contains field names:** Uses the first line of the CSV as headers, which automatically names all extracted fields.

- **Ignore unparseable lines:** Ignores any line that does not correspond to the settings above.

PDF file Input Data settings

PDF Files have a natural, static delimiter in the form of pages, so the options here are interpretation settings for text in the PDF file.

The Input Data settings for PDF files determine how words, lines and paragraphs are detected in the PDF when creating data selections.



Each value represents a fraction of the average font size of text in a data selection, meaning "0.3" represents 30% of the height or width.

- **Word spacing:** Determines the spacing between words. As PDF text spacing is somehow done through positioning instead of actual text spaces, text position is what is used to find new words. This option determines what percentage of the average width of a single character needs to be empty to consider a new word has started. The default value is 0.3, meaning a space is assumed if there is a blank area of 30% of the width of the average character in the font.
- **Line spacing:** Determines the spacing between lines of text. The default value is 1, meaning the space between lines must be equal to at least the average character height.
- **Paragraph spacing:** Determines the spacing between paragraphs. The default value is 1.5, meaning the space between paragraphs must be equal to at least 1.5 times the average character height to start a new paragraph.
- **Magic number:** Determines the tolerance factor for all of the above values. The tolerance is meant to avoid rounding errors. If two values are more than 70% away from each other, they are considered distinct; otherwise they are the same. For example, if two characters have a space of exactly the width of the average character, any space of between 0.7 and 1.43 of this average width is considered one space. A space of 1.44 is considered to be 2 spaces.
- **PDF file color space:** Determines if the PDF is displayed in Color or Monochrome in the Data Viewer. Monochrome display is faster in the Data Viewer. This has no influence on the actual data extraction or the data mapping performance.

Database Input Data settings

Databases all return the same type of information. Therefore the Input Data options for a database refer to the database itself instead of to the data.

The following settings apply to any database or ODBC Data Sample.

- **Connection String:** Displays the connection string used to access the Data Source.
- **Table:** Displays the tables and stored procedures available in the database. The selected table is the one the data is extracted from. Clicking on any of the tables shows the first line of the data in that table.
- **Encoding:** Defines what encoding is used to read the Data Source (US-ASCII, ISO-8859-1, UTF-8, UTF-16, UTF-16BE or UTF-16LE).
- **Browse button** : Opens the **Edit Database configuration** dialog, which can replace the existing database data source with a new one. This is the same as using the **Replace** feature in the Data Samples window.
- **Custom SQL button** : Click to open the SQL Query Designer (see "SQL Query Designer" on page 213) and type in a custom SQL query. If the database supports stored procedures, including inner joins, grouping and sorting, you can use custom SQL to make a selection from the database, using whatever language the database supports.

Text file Input Data settings

Because text files have many different shapes and sizes, there are many options for the input data in these files.

- **Encoding:** Defines what encoding is used to read the Data Source (US-ASCII, ISO-8859-1, UTF-8, UTF-16, UTF-16BE or UTF-16LE).
- **Selection/Text is based on bytes:** Check for text files that use double-bytes characters (resolves width issues in some text files).
- **Add/Remove characters:** Defines the number of characters to add to, or remove from, the head of the data stream. The spin buttons can also increment or decrement the value. Positive values add blank characters while negative values remove characters.
- **Add/Remove lines:** Defines the number of lines to add to, or remove from, the head of the data stream. The spin buttons can also increment or decrement the value. Positive values add blank lines while negative values remove lines.
- **Maximum line length:** Defines the number of columns on a data page. The spin buttons can also increment or decrement the value. The maximum value for this option is 65,535 characters. The default value is 80 characters. You should tune this value to the longest line in your input data. Setting a maximum data line length that greatly exceeds the length of the longest line in your input data may increase execution time.
- **Page delimiter type:** Defines the delimiter between each page of data. Multiples of such pages can be part of a record, as defined by the **Boundaries**.

- **On lines:** Triggers a new page in the Data Sample after a number of lines.
 - **Cut on number of lines:** Triggers a new page after the given number of lines. With this number set to 1, and the Boundaries set to On delimiter, it is possible to create a record for each and every line in the file.
 - **Cut on FF:** Triggers a new page after a Form Feed character.
- **On text:** Triggers a new page in the Data Sample when a specific string is found in a certain location.
 - **Word to find:** Compares the text value with the value in the data source.
 - **Match case:** Activates a case-sensitive text comparison.
 - **Location:** Choose **Selected area** or **Entire width** to use the value of the current data selection as the text value.
 - **Left/Right:** Use the spin buttons to set the start and stop columns to the current data selection (**Selected area**) in the record.
 - **Lines before/after:** This option places the delimiter a certain number of lines before or after the current line. This is useful if the text that triggers the delimiter is not on the first line of each page.
- **Text from right to left:** Sets the writing direction of the data source to right-to-left.
- **Expand tabs to spaces:** Replaces tabs with the given number of spaces.

XML File Input Data settings

For an XML file you can either choose to use the root node, or select an element type, to create a new delimiter every time that element is encountered.

- **Use root element:** Locks the **XML Elements** option to the top-level element. No other boundaries can be set. If there is only one top-level element, there will only be one record.
- **XML elements:** Displays a list containing all the elements in the XML file. Selecting an element causes a new page of data to be created every time an instance of this element is encountered.

Note

The information contained in all of the selected parent nodes will be copied for each

instance of that node. For example, if a client node contains multiple invoice nodes, the information for the client node can be duplicated for each invoice.

The DataMapper only extracts elements for which at least one value or attribute value is defined in the file.

Boundaries

Boundaries are the division between **records**: they define where one record ends and the next record begins; for an explanation see "Record boundaries" on page 117.

CSV or Database file boundaries

Since database data sources are structured the same way as CSV files, the options for these file types are identical.

- **Record limit:** Defines how many records are displayed in the Data Viewer. This does not affect output production; when generating output, this option is ignored. To disable the limit, use the value 0 (zero).
- **Line limit:** Defines the limit of detail lines in any detail table. This is useful for files with a high number of detail lines, which in the DataMapper interface can slow down things. This does not affect output production; when generating output, this option is ignored. To disable the limit, use the value 0 (zero).
- **Trigger:** Defines the type of rule that controls when a boundary is set, creating a new record.
 - **Record(s) per page:** Defines a fixed number of lines in the file that go in each record.
 - **Records:** The number of records (lines, rows) to put in each record.
 - **On change:** Defines a new record when a specific field (**Field name**) has a new value.
 - **Field name:** Displays the fields in the top line. The boundaries are set on the selected field name.
 - **On script:** Defines the boundaries using a custom JavaScript. For more information see "Setting boundaries using JavaScript" on page 257.

- **On field value:** Sets a boundary on a specific field value.
 - **Field name:** Displays the fields in the top line. The value of the selected field is compared with the **Expression** below to create a new boundary.
 - **Expression:** Enter the value or **Regular Expression** to compare the field value to.
 - **Use Regular Expression:** Treats the **Expression** as a regular expression instead of static text. For more information on using **Regular Expressions** (regex), see the [Regular-Expressions.info Tutorial](#).

PDF file boundaries

For a PDF file, Boundaries determine how many pages are included in each record. You can set this up in one of three ways: by giving a static number of pages; by checking a specific area on each page for text changes, specific text, or the absence of text; or by using an advanced script.

- **Record limit:** Defines how many records are displayed in the Data Viewer. To disable the limit, use the value 0 (zero).
- **Trigger:** Defines the type of rule that controls when a boundary is set, creating a new record.
 - **On page:** Defines a boundary on a static number of pages.
 - **Number of pages:** Defines how many pages go in each record.
 - **On text:** Defines a boundary on a specific text comparison.
 - **Start coordinates (x,y):** Defines the left and top coordinates of the data selection to compare with the text value.
 - **Stop coordinates (x,y):** Defines the right and bottom coordinates.
 - **Use Selection:** Select an area in the Data Viewer and click the **Use selection** button to set the start and stop coordinates to the current data selection.

Note

In a PDF file, all coordinates are in millimeters.

- **Times condition found:** When the boundaries are based on the presence of specific text, you can specify after how many instances of this text the

boundary can be effectively defined. For example, if a string is always found on the first and on the last page of a document, you could specify a number of occurrences of 2. This way, there is no need to inspect other items for whether it is on the first page or the last page. Having found the string two times is enough to set the boundary.

- **Pages before/after:** Defines the boundary a certain number of pages before or after the current page. This is useful if the text triggering the boundary is not located on the first page of the record.
- **Operator:** Selects the type of comparison (for example, "contains").
- **Word to find:** Compares the text value with the value in the data source.
- **Match case:** Makes the text comparison case-sensitive.

Text file boundaries

For a text file, Boundaries determine how many 'data pages' are included in each record. These don't have to be actual pages, as is the case with PDF files. The data page delimiters are set in the "Text file Input Data settings" on page 205.

- **Record limit:** Defines how many records are displayed in the Data Viewer. This does not affect output production; when generating output, this option is ignored. To disable the limit, use the value 0 (zero).
- **Selection/Text is based on bytes:** Select this option for text records with fixed width fields whose length is based on the number of bytes and not the number of characters.
- **Trigger:** Defines the type of rule that controls when a boundary is set, creating a new record.
 - **On delimiter:** Defines a boundary on a static number of pages.
 - **Occurrences:** The number of times that the delimiter is encountered before fixing the boundary. For example, if you know that your documents always have four pages delimited by the FF character, you can set the boundaries after every four delimiters.
 - **On text:** Defines a boundary on a specific text comparison.
 - **Location:**
 - **Selected area:**
 - **Select the area** button: Uses the value of the current data selection as the text value. Making a new selection and clicking on Select the area will redefine the location.

- **Left/Right:** Defines where to find the text value in the row.
- **Top/Bottom:** Defines the start and end row of the data selection to compare with the text value.
- **Entire width:** Ignores the column values and compares using the whole line.
- **Entire height:** Ignores the row values and compares using the whole column.
- **Entire page:** Compares the text value on the whole page. Only available with `contains`, `not contains`, `is empty` and `is not empty` operators.
- **Times condition found:** When the boundaries are based on the presence of specific text, you can specify after how many instances of this text the boundary can be effectively defined. For example, if a string is always found on the first and on the last page of a document, you could specify a number of occurrences of 2. This way, there is no need to inspect other items for whether it is on the first page or the last page. Having found the string two times is enough to set the boundary.
- **Delimiters before/after:** Defines the boundary a certain number of data pages before or after the current data page. This is useful if the text triggering the boundary is not located on the first data page of the record.
- **Operator:** Selects the type of comparison (for example, "contains").
- **Word to find:** Compares the text value with the value in the data source.
- **Use selected text** button: copies the text in the current selection as the one to compare to it.
- **Match case:** Makes the text comparison case-sensitive.
- **On script:** Defines the boundaries using a custom JavaScript. For more information see "Setting boundaries using JavaScript" on page 257.

XML file boundaries

The delimiter for an XML file is a node. The Boundaries determine how many of those nodes go in one record. This can be a specific number, or a variable number if the boundary is to be set when the content of a specific field or attribute within a node changes (for example when the `invoice_number` field changes in the `invoice` node).

- **Record limit:** Defines how many records are displayed in the Data Viewer. This does not affect output production; when generating output, this option is ignored. To disable the limit, use the value 0 (zero).
- **Trigger:** Defines the type of rule that controls when a boundary is set, creating a new record.
 - **On Element:** Defines a new record on each new instance of the XML element selected in the Input Data settings.
 - **Occurrences:** The number of times that the element is encountered before fixing the boundary.
 - **On Change:** Defines a new record when a specific field or attribute in the XML element has a new value.
 - **Field:** Displays the fields and (optionally) attributes in the XML element. The value of the selected field determines the new boundaries.
 - **Also extract element attributes:** Check this option to include attribute values in the list of content items that can be used to trigger a boundary.

Data samples

The Data Sample area displays a list of all the imported Data Samples that are available in the current data mapping configuration. As many Data Samples as necessary can be imported to properly test the configuration.


Only one of the data samples - the active data sample - is shown in the Data Viewer.





A number of buttons let you manage the Data Samples.

In addition to using the buttons listed below, you can right-click a file to bring up the context menu, which offers the same options plus the **Copy** and **Paste** options.

Tip





Data samples can be copied and pasted **to** and **from** the Settings pane using Windows File Explorer.

- **Add** : Add a new Data Sample from an external data source. The new Data Sample will need to be of the same data type as the current one. For example, you can only add PDF files to a PDF data mapping configuration. Multiple files can be added simultaneously.

- **Delete** : Remove the current Data Sample from the data mapping configuration.
- **Replace** : Open a Data Sample and replace it with the contents of a different data source.
- **Reload** : Reload the currently selected Data Sample and any changes that have been made to it.
- **Set as Active** : Activates the selected Data Sample. The active data sample is shown in the Data Viewer after it has gone through the **Preprocessor** step as well as the **Input Data** and **Boundary** settings.

External JS Libraries

Right-clicking in the box brings up a control menu, with the same options as are available through the buttons on the right.

- **Add** : Add a new external library. Use the standard **Open** dialog to browse and open the .js file.
- **Delete** : Remove the currently selected library from the data mapping configuration.
- **Replace** : Open a library and replace it with the contents of a different js file.
- **Reload** : Reload the currently selected library and any changes that have been made to it.

Default Data Format

The Default Data Format settings defined here apply to any new extraction in made in the current data mapping configuration. Any format already defined for an existing field remains untouched.

It is also possible to set a default format for dates and currencies in the user preferences ("Datamapper preferences" on page 703).

Specific settings for a field that contains extracted data are made via the properties of the Extract step that the field belongs to (see "Editing fields" on page 158).

- **Negative Sign Before** : A negative sign will be displayed before any negative value.
- **Decimal Separator** : Set the decimal separator for a numerical value.
- **Thousand Separator** : Set the thousand separator for a numerical value.
- **Currency Sign** : Set the currency sign for a currency value.
- **Date Format** : Set the date format for a date value.


- **Date Language** : Set the date language for a date value (ex: If English is selected, the term May will be identified as the month of May).
- **Treat empty as 0** : A numerical empty value is treated as a 0 value.

Note

Default data formats tell the DataMapper how certain types of data are formatted **in the data source**. They don't determine how these data are formatted in the Data Model or in a template. In the Data Model, data are converted to the native data type. Dates, for example, are converted to a DateTime object in the Data Model, and will always be shown as "year-month-day" plus the time stamp, for example: 2012-04-11 12.00 AM.

SQL Query Designer

The SQL Query Designer is used to design a custom SQL query to pull information from a database. It can be opened via the Settings pane when extracting data from a database.



- **Tables**: Lists all tables and stored queries in the database.
- **Custom Query**: Displays the query that retrieves information from a database. Each database type has their own version of the SQL query language. To learn how to build your own query, please refer to your database's user manual.
- **Test Query** button : Click to test the custom query to ensure it will retrieve the appropriate information.
- **Results**: Displays the result of the SQL query when clicking on **Test Query**.

Steps pane

The Steps tab displays the data mapping workflow: the process that prepares and extracts data. The process contains multiple distinct steps and is run for each of the records in the source data. For more information about the steps and how to use them, please refer to [Steps](#) and "Data mapping workflow" on page 113.

Moving a step

To rearrange steps, simply drag & drop them somewhere else on the dotted line in the Steps pane.

Alternatively you can right-click on a step and select **Cut Step** or use the Cut  button in the **Toolbar**. If the step is **Repeat** or **Condition**, all steps under it will also be placed in the clipboard. To place the step at its destination, right-click the step in the position before the desired location and click **Paste Step**, or use the Paste  button in the toolbar.

Viewing step details



Hovering over the task shows a tooltip that displays some of the details of that step. To see all details for a step, click on the step and take a look at the Step properties pane ("Step properties pane" on the next page).

Clicking on any Extract step in the Steps pane highlights any area in the Data Viewer from which it extracts data.

You can also click on the Preprocessor step to select all the steps in the workflow to show a complete map of all the extracted data.

Window controls




The following controls appear at the top of the Steps pane:

- **Zoom In (CTRL +)** : Click to zoom in by increments of 10%
- **Zoom Out (CTRL -)** : Click to zoom out by increments of 10%

Contextual menu

You can access the contextual menu using a right-click anywhere inside the **Steps** pane.

- **Add a Step**: Adds a step to the process. More options are available when a **Repeat** or a **Condition** step is selected:
 - **Add Step in Repeat**: Adds a step to a **Repeat** loop.
 - **Add Step in True**: Adds a step to the True branch of a **condition** step.
 - **Add Step in False**: Adds a step under the False branch of a **condition** step.
 - **Add Multiple Conditions Step**: Adds a **Multiple Conditions** step.

- **Add Case Step:** Adds a Case condition under the selected **Multiple Conditions** step.
- **Ignore Step:** Click to set the step to be ignored (aka disabled). Disabled steps are grayed and do not run, neither in the DataMapper nor when the data mapping configuration is executed in Workflow. However, they can still be modified normally.
- **Delete Step:** To remove a step, right-click on it and select **Delete step** from the contextual menu or use the Delete  button in the **Toolbar**. If the step to be deleted is **Repeat** or **Condition**, all steps under it will also be deleted.
- **Copy/Paste Step:** To copy a step, right-click on it and select **Copy Step** or use the  button in the **Toolbar**. If the step is **Repeat** or **Condition**, all steps under it will also be placed in the clipboard. To paste the copied step at its destination, right-click the step in the position before the desired location and select **Paste Step**, or use the  button in the **Toolbar**.

Step properties pane

The Step Properties pane is used to adjust the properties of each step in the process. The pane is divided in a few subsections depending on the step and the data type. It always contains a subsection to name and document the selected step.

Note

Step properties may also depend on the data sample's file type.

- "Preprocessor step properties" on the facing page
- "Settings for location-based fields in a Text file" on page 221
- "Text and PDF Files" on page 226
- "Text and PDF Files" on page 233
- "Condition step properties" on page 238
- "Left operand, Right operand" on page 241
- "Text file" on page 244
- "JavaScript " on page 249

Preprocessor step properties

The **Preprocessor** step does not run for every record in the source data. It runs once, at the beginning of the extraction workflow, before anything else; see "Preprocessor step" on page 140.

The properties described below become visible in the Step properties pane when the Preprocessor step is selected in the Steps pane.

Description

This subsection is collapsed by default in the interface, to give more screen space to other important parts.

Name: The name of the step. This name will be displayed on top of the step's icon in the **Steps** pane.

Comments: The text entered here will be displayed in the tooltip that appears when hovering over the step in the **Steps** pane.

Fixed Automation Properties

The **Fixed automation properties** subsection lists all the fixed properties available from the PlanetPress Workflow automation module. These properties are equivalent to data available within the PlanetPress Workflow process. For each property, the following is available:

- **Name:** A read-only field displaying the name of the property.
- **Scope:** A read-only field indicating that the scope of the property is **Automation**.
- **Type:** A read-only field indicating the data type for each property.
- **Default Value:** Enter a default value for the property. This value is overwritten by the actual value coming from PlanetPress Workflow when the data mapping configuration is run using the **Execute Data Mapping** task.

There are currently the following automation properties available:

- **JobInfoX:** These properties are the equivalent of the JobInfo values available in the PlanetPress Workflow process. They can be set using the Set Job Info and Variables task. To access these properties inside of any JavaScript code within the Data Mapping

Configuration, use the automation.jobInfos.JobInfoX (where X is the job info number, from 0 to 9).

- **OriginalFilename:** This property contains the original file name that was captured by the PlanetPress Workflow process and is equivalent to the %o variable in the process. To access these property inside of any JavaScript code within the Data Mapping Configuration, use automation.properties.OriginalFilename.
- **ProcessName:** This property contains the name of the process that is currently executing the data mapping configuration and is equivalent to the %w variable in the process. To access this property inside of any JavaScript code within the Data Mapping Configuration, use automation.properties.ProcessName.
- **TaskIndex:** This property contains the index (position) of the task inside the process that is currently executing the data mapping configuration but it has no equivalent in PlanetPress Workflow. To access this property inside of any JavaScript code within the Data Mapping Configuration, use automation.properties.ProcessName.

Properties

The **Properties** subsection is used to create specific properties that are used throughout the workflow. Properties can be accessed through some of the interface elements such as the **Condition** and **Repeat** step properties, or in scripts, through the "DataMapper Scripts API" on page 252.

Note

Properties are evaluated in the order they are placed in the list, so properties can use the values of previously defined properties in their expression.

- **Name:** The name of the property used to refer to its value.
- **Scope:** What this property applies to:
 - **Entire Data:** These properties are static properties that cannot be changed once they have been set, in other words they are **Global constants**.
 - **Each Record:** These properties are evaluated and set at the beginning of each Source Record. These properties can be modified once they have been set, but are always reset at the beginning of each Source Record.

- **Automation variable:** These properties initialize variables coming from the PlanetPress Workflow automation tool. The name of the property needs to be the same as the variable name in Workflow, and they can be either a Local variable or a Global variable. For either one, only the actual name is to be used, so for `{MyLocalVar}` use only `MyLocalVar`, and for `{global.MyGlobalVar}` use `MyGlobalVar`. If a global and a local variable have the same name (`{myvar}` and `{global.myvar}`), the local variable's value is used and the global one is ignored. To access a workflow variable inside of any JavaScript code within the Data Mapping Configuration, use `automation.variables.variablename`
- **Type:** The data type of the property. For more information see "Data types" on page 168.
- **Default Value:** The initial value of the property. This is a JavaScript expression. See "DataMapper Scripts API" on page 252.

Note

Entire **Data Properties** are evaluated before anything else, such as **Preprocessors**, **Delimiters** and **Boundaries** in the Settings pane (see "Data source settings" on page 115). This means these properties cannot read information from the data sample or from any records. They are mostly useful for static information such as folder locations or server addresses.

Preprocessor

The **Preprocessor** subsection defines what preprocessor tasks are performed on the data file before it is handed over to the data mapping workflow. Preprocessor tasks can modify the data file in many ways, and each task runs in turn, using the result of the previous one as an input.

- **Name:** The name to identify the **Preprocessor** task.
- **Type:** The type of **Preprocessor** task. Currently there is only one type available: script.

Preprocessor definition

- **Expression:** Enter the JavaScript code to be performed on the data file. See "DataMapper Scripts API" on page 252.

Extract step properties

The **Extract** step takes information from the data source and places it in the record set that is the result of the extraction workflow. For more information see "Extract step" on page 142 and "Extracting data" on page 118.

Description

This subsection is collapsed by default in the interface, to give more screen space to other important parts.

Name: The name of the step. This name will be displayed on top of the step's icon in the **Steps** pane.

Comments: The text entered here will be displayed in the tooltip that appears when hovering over the step in the **Steps** pane.

Extraction Definition

- **Data Table:** Defines where the data will be placed in the extracted record. The root table is record, any other table inside the record is a detail table. For more information see "Extracting transactional data" on page 124.
- **Append values to current record:** When the **Extract** step is inside a loop, check this to ensure that the extraction will be done in the same **detail table** as any previous extractions within the same loop. This ensures that, if multiple extracts are present, only one detail table is created.

Field Definition

The following field definition settings are identical for all fields.

- **Field List:** The **Field List** displays each of the single fields that belong to the selected step in a drop-down. Fields can be re-ordered and re-named within the Order and rename fields dialog (see "Order and rename fields dialog" on page 224). Select one of the fields to make further settings for that field.
- **Add Unique ID to extraction field:** Check to add a unique numerical set of characters to the end of the extracted value. This ensures no two values are identical in this field in the record set.

- **Mode:** Determines the origin of the data. Fields always belong to an Extract step, but they don't necessarily contain extracted data. See "Fields" on page 155 for more information.
 - **Location:** The contents of the data selection determine the value of the extracted field. The settings for location-based fields are listed separately, per file type:
 - "Settings for location-based fields in a Text file" on the next page
 - "Settings for location-based fields in a PDF File" on the next page
 - "Settings for location-based fields in CSV and Database files" on page 222
 - "Settings for location-based fields in an XML File" on page 223
 - **JavaScript :** The result of the JavaScript Expression written below the drop-down will be the value of the extracted field. If the expression contains multiple lines, the last value attribution (variable = "value";) will be the value. See [DataMapper API](#).
 - **Use JavaScript Editor:** Click to display the [Script Editor](#) dialog.
 - **Use selected text:** Inserts the text in the current data selection in the JavaScript Expression. If multiple lines or elements are selected, only the first one is used.
 - **Use selection:** Click to use the value of the current data selection for the extraction.

Note

If the selection contains multiple lines, only the first line is selected.

- **Properties:** The value of the property selected below will be the value of the selected field.
 - **Property:** This drop-down lists all the currently defined properties (including system properties).
Custom properties can be defined in the Preprocessor step; see "Preprocessor step" on page 140.
For an explanation of the objects to which the properties belong, see "DataMapper Scripts API" on page 252.
 - **Choose a property** button: Click this button to open a filter dialog that lets you find a property based on the first few letters that you type.

- **Type:** The data type of the selected data; see "Data types" on page 168. Make sure that the data format that the DataMapper expects matches the actual format of the data in the data source; see "Data Format" on page 223.

Settings for location-based fields in a Text file

- **Left:** Defines the start of the data selection to extract.
- **Right:** Defines the end of the data selection to extract.
- **Top offset:** The vertical offset from the current pointer location in the Data Viewer).
- **Height:** The height of the selection box. When set to 0, this instructs the DataMapper to extract all lines starting from the given position until the end of the record and store them in a single field.
- **Use selection:** Click to use the value (Left, Right, Top offset and Height) of the current data selection (in the Data Viewer) for the extraction.

Note

If the selection contains multiple lines, only the first line is extracted.

- **Post Function:** Enter a JavaScript expression to be run after the extraction. A Post function script operates directly on the extracted data, and its results replace the extracted data. For example, the Post function script `replace("-", "");` would replace the first dash character that occurs inside the extracted string.
 - **Use JavaScript Editor:** Click to display the [Script Editor](#) dialog.
- **Trim:** Select to trim empty characters at the beginning or the end of the field.
- **Concatenation string:**
- **Split:** Separate the selection into individual fields based on the **Concatenation string** defined above.

Settings for location-based fields in a PDF File

These are the settings for location-based fields in a PDF file.

- **Left:** Defines the start of the data selection to extract.
- **Right:** Defines the end of the data selection to extract.
- **Top offset:** The vertical offset from the current pointer location in the Data Sample (Viewer).
- **Height:** The height of the selection box.
- **Use selection:** Click to use the value (Left, Right, Top offset and Height) of the current data selection for the extraction.

Note

If the selection contains multiple lines, the lines are by default joined and extracted into one field. To split the lines, select the option Split lines (see below).

- **Post Function:** Enter a JavaScript expression to be run after the extraction. For example `replace("-", "")` would replace a single dash character inside the extracted string.
- **Trim:** Select to trim empty characters at the beginning or the end of the field.
- **Type:** The data type of the selected data; see "Data types" on page 168. If the selected data is split (see below), this setting is applied to the first extracted field. Make sure that the data format that the DataMapper expects matches the actual format of the data in the data source; see "Data Format" on the next page.
- **Split:**
 - **Split lines:** Separate a multi-line selection into individual fields .
 - **Join lines:** Join the lines in the selection with the Concatenation string defined below.
 - **Concatenation string:** The (HTML) string used to concatenate lines when they are joined.

Settings for location-based fields in CSV and Database files

These are the settings for location-based fields in CSV and Database files.

- **Column:** Drop-down listing all fields in the Data Sample, of which the value will be used.
- **Top offset:** The vertical offset from the current pointer location in the Data Sample (Viewer).

- **Use selection:** Click to use the value of the current data selection for the extraction.

Note

If the selection contains multiple lines, only the first line is selected.

- **Post Function:** Enter a JavaScript expression to be run after the extraction. For example `replace("-", "")` would replace a single dash character inside the extracted string.
 - **Use JavaScript Editor:** Click to display the [Script Editor](#) dialog.
- **Trim:** Select to trim empty characters at the beginning or the end of the field.

Settings for location-based fields in an XML File

These are the settings for location-based fields in an XML file.

- **XPath:** The path to the XML field that is extracted.
 - **Use selection:** Click to use the value of the current data selection for the extraction.

Note

If the selection contains multiple lines, only the first line is selected.

- **Post Function:** Enter a JavaScript expression to be run after the extraction. For example `replace("-", "")` would replace a single dash character inside the extracted string.
 - **Use JavaScript Editor:** Click to display the [Script Editor](#) dialog.
- **Trim:** Select to trim empty characters at the beginning or the end of the field.

Data Format

Format settings can be defined in three places: in the user preferences ("Datamapper preferences" on page 703), the current data mapping configuration ("Data format settings" on page 118) and per field via the Step properties.

Any format settings specified per field are always used, regardless of the user preferences or data source settings.

Note

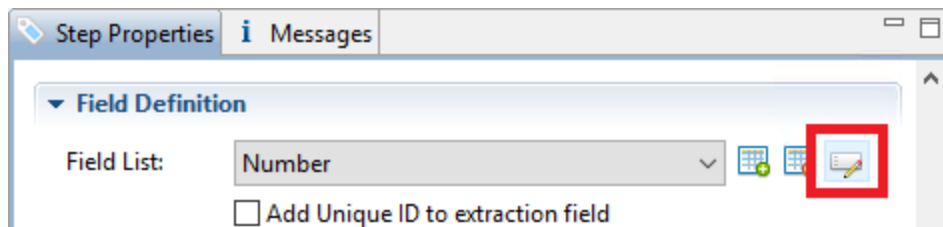
Data format settings tell the DataMapper how certain types of data are formatted **in the data source**. They don't determine how these data are formatted in the Data Model or in a template. In the Data Model, data are converted to the native data type. Dates, for example, are converted to a DateTime object in the Data Model, and will always be shown as "year-month-day" plus the time stamp, for example: 2012-04-11 12.00 AM.

- **Negative Sign Before** : A negative sign will be displayed before any negative value.
- **Decimal Separator** : Set the decimal separator for a numerical value.
- **Thousand Separator** : Set the thousand separator for a numerical value.
- **Currency Sign** : Set the currency sign for a currency value.
- **Date Format** : Set the date format for a date value.
- **Date Language** : Set the date language for a date value (ex: If English is selected, the term May will be identified as the month of May).
- **Treat empty as 0** : A numerical empty value is treated as a 0 value.

Order and rename fields dialog

The **Order and rename fields** dialog displays the extracted fields in the currently selected **Extract** step.

To open it, first select an Extract step on the Steps pane. Then, on the Step properties pane, under Field Definition, click the **Order and Rename Fields** button  next to the Field List dropdown.






Field extractions are executed from top to bottom.

In **JavaScript-based** fields, it is possible to refer to previously extracted fields if they are extracted higher in this list or in previous **Extract** steps in the extraction workflow.

- **Name:** The name of the field. Click the field name and enter a new name to rename the field.

Note

If you intend to use the field names as metadata in a PlanetPress Workflow process, do not add spaces to field names, as they are not permitted in metadata field names.

- **Value:** Displays the value of the extract field in the current Record.
- **Remove** button : Click to remove the currently selected field.
- **Move Up** button : Click to move the selected field up one position.
- **Move Down** button : Click to move the selected field down one position.

Action step properties

The **Action** step can run multiple specific actions one after the other in order; see "Action step" on page 149 for more information.

The properties of an Action step become visible in the Step properties pane when the Action step is selected on the Steps pane.

Description

This subsection is collapsed by default in the interface, to give more screen space to other important parts.

Name: The name of the step. This name will be displayed on top of the step's icon in the **Steps** pane.

Comments: The text entered here will be displayed in the tooltip that appears when hovering over the step in the **Steps** pane.

Actions

This subsection lists all actions executed by the step, and their types.

- **Name:** A name by which to refer to the action. This name has no impact on functionality.
- **Type:**
 - **Set property:** Sets the value of a record property which was created in the Preprocessor step (see "Preprocessor step" on page 140).
 - **Run JavaScript :** Runs a JavaScript expression, giving much more flexibility over the extraction process.
 - **Stop Processing Record:** When this option is selected, the extraction workflow stops processing the current record and moves on to the next one.
If fields were already extracted prior to encountering the Action step, then those fields are stored as usual.
If no fields were extracted prior to encountering the Action step, then no trace of the record is saved in the database at run time.

Set Property

Text and PDF Files

- **Property:** Displays a list of record properties set in the Preprocessor step (see "Preprocessor step" on page 140).
- **Type:** Displays the type of the property. This is a read-only field.
- **Based on:** Determines the origin of the data.
 - **Location:** The contents of the data selection set below will be the value of the extracted field. The data selection settings are different depending on the data sample type.
 - **Left:** Defines the start of the data selection to extract
 - **Right:** Defines the end of the data selection to extract
 - **Top offset:** The vertical offset from the current pointer location in the Data Sample (**Viewer**).
 - **Height:** The height of the selection box.
 - **Use selection:** Click to use the value of the current data selection for the extraction.

Note

If the selection contains multiple lines, only the first line is selected.

- **Trim:** Select to trim empty characters at the beginning or the end of the field
- **JavaScript :** The result of the JavaScript **Expression** written below the drop-down will be the value of the extracted field. If the expression contains multiple lines, the last value attribution (variable = "value";) will be the value. See "DataMapper Scripts API" on page 252.
 - **Expression:** The JavaScript expression to run.
 - **Use JavaScript Editor:** Click to display the Edit Script dialog (see "Using scripts in the DataMapper" on page 255 and "DataMapper Scripts API" on page 252).
 - **Use selected text:** Inserts the text in the current data selection in the JavaScript Expression. If multiple lines or elements are selected, only the first one is used.
 - **Use selection:** Click to use the value of the current data selection for the extraction.

Note

If the selection contains multiple lines, only the first line is selected.

- **Data Format:** Data format settings tell the DataMapper how certain types of data are formatted **in the data source**. Make sure that this format matches the actual format of the data in the data source.
 - **Negative Sign Before :** A negative sign will be displayed before any negative value.
 - **Decimal Separator :** Set the decimal separator for a numerical value.
 - **Thousand Separator :** Set the thousand separator for a numerical value.
 - **Currency Sign :** Set the currency sign for a currency value.
 - **Date Format :** Set the date format for a date value.

- **Date Language** : Set the date language for a date value (ex: If English is selected, the term May will be identified as the month of May).
- **Treat empty as 0** : A numerical empty value is treated as a 0 value.

CSV and Database Files

- **Property**: Displays a list of record properties set in the Preprocessor step (see "Preprocessor step" on page 140).
- **Type**: Displays the type of the property. Read only field.
- **Based on**: Determines the origin of the data.
 - **Location**: The contents of the data selection set below will be the value of the extracted field. The data selection settings are different depending on the data sample type.
 - **Column**: Drop-down listing all fields in the Data Sample, of which the value will be used.
 - **Top offset**: The vertical offset from the current pointer location in the Data Sample (**Viewer**).
 - **Use selection**: Click to use the value of the current data selection for the extraction.

Note

If the selection contains multiple lines, only the first line is selected.

- **Trim**: Select to trim empty characters at the beginning or the end of the field
- **JavaScript** : The result of the JavaScript **Expression** written below the drop-down will be the value of the extracted field. If the expression contains multiple lines, the last value attribution (variable = "value";) will be the value. See "DataMapper Scripts API" on page 252.
 - **Expression**: The JavaScript expression to run.
 - **Use JavaScript Editor**: Click to display the Edit Script dialog (see "Using scripts in the DataMapper" on page 255 and "DataMapper Scripts API" on page 252).

- **Use selected text:** Inserts the text in the current data selection in the JavaScript Expression. If multiple lines or elements are selected, only the first one is used.
- **Use selection:** Click to use the value of the current data selection for the extraction.

Note

If the selection contains multiple lines, only the first line is selected.

- **Data Format:** Data format settings tell the DataMapper how certain types of data are formatted **in the data source**. Make sure that this format matches the actual format of the data in the data source.
 - **Negative Sign Before :** A negative sign will be displayed before any negative value.
 - **Decimal Separator :** Set the decimal separator for a numerical value.
 - **Thousand Separator :** Set the thousand separator for a numerical value.
 - **Currency Sign :** Set the currency sign for a currency value.
 - **Date Format :** Set the date format for a date value.
 - **Date Language :** Set the date language for a date value (ex: If English is selected, the term May will be identified as the month of May).
 - **Treat empty as 0 :** A numerical empty value is treated as a 0 value.

XML File

- **Property:** Displays a list of Source Record properties set in the Preprocessor step (see "Preprocessor step" on page 140).
- **Type:** Displays the type of the Source Record property. Read only field.
- **Based on:** Determines the origin of the data.
 - **Location:** The contents of the data selection set below will be the value of the extracted field. The data selection settings are different depending on the data sample type.

- **XPath:** The path to the XML field that is extracted.
 - **Use selection:** Click to use the value of the current data selection for the extraction.

Note

If the selection contains multiple lines, only the first line is selected.

- **Trim:** Select to trim empty characters at the beginning or the end of the field
- **JavaScript :** The result of the JavaScript **Expression** written below the drop-down will be the value of the extracted field. If the expression contains multiple lines, the last value attribution (variable = "value";) will be the value. See "DataMapper Scripts API" on page 252.
 - **Expression:** The JavaScript expression to run.
 - **Use JavaScript Editor:** Click to display the Edit Script dialog (see "Using scripts in the DataMapper" on page 255 and "DataMapper Scripts API" on page 252).
 - **Use selected text:** Inserts the text in the current data selection in the JavaScript Expression. If multiple lines or elements are selected, only the first one is used.
 - **Use selection:** Click to use the value of the current data selection for the extraction.

Note

If the selection contains multiple lines, only the first line is selected.

- **Data Format:** Data format settings tell the DataMapper how certain types of data are formatted **in the data source**. Make sure that this format matches the actual format of the data in the data source.
 - **Negative Sign Before :** A negative sign will be displayed before any negative value.
 - **Decimal Separator :** Set the decimal separator for a numerical value.
 - **Thousand Separator :** Set the thousand separator for a numerical value.
 - **Currency Sign :** Set the currency sign for a currency value.
 - **Date Format :** Set the date format for a date value.
 - **Date Language :** Set the date language for a date value (ex: If English is selected, the term May will be identified as the month of May).
 - **Treat empty as 0 :** A numerical empty value is treated as a 0 value.

Run JavaScript

Running a JavaScript expression offers many possibilities. The script could, for example, set record properties and field values using advanced expressions and complex mathematical operations and calculations.

- **Expression:** The JavaScript expression to run (see "DataMapper Scripts API" on page 252).
- **Use JavaScript Editor:** Click to display the Edit Script dialog.
- **Use selected text:** Inserts the text in the current data selection in the JavaScript Expression. If multiple lines or elements are selected, only the first one is used.
- **Use selection:** Click to use the value of the current data selection.

Note

If the selection contains multiple lines, only the first line is selected.

Repeat step properties

The **Repeat** step adds a loop to the extraction workflow; see "Steps" on page 140 and "Extracting transactional data" on page 124.

The properties described below become visible in the Step properties pane when the Repeat step is selected in the Steps pane.

Description

This subsection is collapsed by default in the interface, to give more screen space to other important parts.

Name: The name of the step. This name will be displayed on top of the step's icon in the **Steps** pane.

Comments: The text entered here will be displayed in the tooltip that appears when hovering over the step in the **Steps** pane.

Repeat Definition

- **Repeat type:**

- **While statement is true:** The loop executes while the statement below is true. The statement is evaluated after the loop so the loop will always run at least once.
- **Until statement is true:** The loop executes until the statement below is true. The statement is evaluated before the loop so the loop may not run at all.
- **Until no more elements (for Text, CSV, Database and PDF files only):** The loop executes as long as there are elements left as selected below.
- **For Each (for XML files only):** The loop executes for all nodes on a specified level.

Note

When using an XML **For Each** loop, it is not necessary to skip to the repeating node or to have a **Go to** step to jump to each sibling, as this loop takes care of it automatically.

- **Maximum iterations on each line:** Defines the maximum number of iterations occurring at the same position. This expression is evaluated once when entering the loop. The value returned by the expression must be an integer higher than 0.
 - **Use JavaScript Editor:** Click to display the Edit Script dialog.

Rule Tree

The **Rule tree** subsection displays the full combination rules (defined below under **Condition**) as a tree, which gives an overview of how the conditions work together as well as the result for each of these conditions for the current record or iteration.

Condition

First, the **Condition List** displays the conditions in list form, instead of the tree form above. Three buttons are available next to the list:

- **Add condition:** Click to create a new condition in the list. This will always branch the current condition as an "AND" operator.
- **Delete condition:** Delete the currently selected condition.
- To rename a **Condition**, double click on its name from the **Rule tree** subsection .

Conditions are made by comparison of two operands using a specific **Operator**.

Note

Both the **Left** and **Right** operands have the same properties.

Text and PDF Files

- **Based On:**
 - **Position:** The data in the specified position for the comparison.
 - **Left:** The start position for the data selection. Note that conditions are done on the current line, either the current cursor position, or the current line in a **Repeat** step.
 - **Right:** The end position for the data selection.
 - **Top offset:** The vertical offset from the current pointer location in the Data Sample (**Viewer**).
 - **Height:** The height of the selection box.
 - **Use Selection:** Click to use the value of the current data selection for the

extraction.

- **Trim**: Select to trim empty characters at the beginning or the end of the field.
- **Value**: A specified static text value.
 - **Value**: The text value to use in the comparison.
 - **Use selected text**: Uses the text in the current data selection as the **Value**. If multiple lines or elements are selected, only the first one is used.
- **Field**: The contents of a specific field in the **Extracted Record**.
 - **Field**: The **Extracted Record** field to use in the comparison.
- **JavaScript** : The result of a JavaScript **Expression**.
 - **Expression**: The JavaScript line that is evaluated. Note that the last value attribution to a variable is the one used as a result of the expression.
 - **Use JavaScript Editor**: Click to display the [Edit Script](#) dialog.
 - **Use selected text**: Inserts the text in the current data selection in the JavaScript **Expression**. If multiple lines or elements are selected, only the first one is used.
- **Data Property**: The value of a data-level property set in the Preprocessor step (see "Preprocessor step" on page 140).
- **Record Property**: One of the local variables that you can create and that are reset for each document as opposed to data variables that are global because they are initialized only once at the beginning of each job.
- **Automation Property**: The current value of a Document-level property set in the Preprocessor step (see "Preprocessor step" on page 140).
- **Extractor Property**: The value of an internal extractor variable:
 - **Counter**: The value of the current counter iteration in a **Repeat** step.
 - **Vertical Position**: The current vertical position on the page, either in Measure (PDF) or Line (Text and CSV).
- **Operators**:
 - **is equal to**: The two specified value are identical for the condition to be **True**.
 - **contains**: The first specified value contains the second one for the condition to be **True**.

- **is less than:** The first specified value is smaller, numerically, than the second value for the condition to be **True**.
- **is greater than:** The first specified value is larger, numerically, than the second value for the condition to be **True**.
- **is empty:** The first specified value is empty. With this operator, there is no second value.
- **Invert condition:** Inverts the result of the condition. For instance, **is empty** becomes **is not empty**.

CSV and Database Files

- **Based On:**
 - **Position:** The data in the specified position for the comparison.
 - **Column:** Drop-down listing all fields in the Data Sample, of which the value will be used.
 - **Top offset:** The vertical offset from the current pointer location in the Data Sample (**Viewer**).
 - **Use Selection:** Click to use the value of the current data selection for the extraction.
 - **Trim:** Select to trim empty characters at the beginning or the end of the field.
 - **Value:** A specified static text value.
 - **Value:** The text value to use in the comparison.
 - **Use selected text:** Uses the text in the current data selection as the **Value**. If multiple lines or elements are selected, only the first one is used.
 - **Field:** The contents of a specific field in the **Extracted Record**.
 - **Field:** The **Extracted Record** field to use in the comparison.
 - **JavaScript :** The result of a JavaScript **Expression**.
 - **Expression:** The JavaScript line that is evaluated. Note that the last value attribution to a variable is the one used as a result of the expression.
 - **Use JavaScript Editor:** Click to display the [Edit Script](#) dialog.

- **Use selected text:** Inserts the text in the current data selection in the JavaScript **Expression**. If multiple lines or elements are selected, only the first one is used.
- **Data Property:** The value of a data-level property set in the **Preprocessor** step.
- **Record Property:** One of the local variables that you can create and that are reset for each document as opposed to data variables that are global because they are initialized only once at the beginning of each job.
- **Automation Property:** The current value of a Document-level property set in the **Preprocessor** step.
- **Extractor Property:** The value of an internal extractor variable:
 - **Counter:** The value of the current counter iteration in a **Repeat** step.
 - **Vertical Position:** The current vertical position on the page, either in Measure (PDF) or Line (Text and CSV).
- **Operators:**
 - **is equal to:** The two specified value are identical for the condition to be **True**.
 - **contains:** The first specified value contains the second one for the condition to be **True**.
 - **is less than:** The first specified value is smaller, numerically, than the second value for the condition to be **True**.
 - **is greater than:** The first specified value is larger, numerically, than the second value for the condition to be **True**.
 - **is empty:** The first specified value is empty. With this operator, there is no second value.
 - **Invert condition:** Inverts the result of the condition. For instance, **is empty** becomes **is not empty**.

XML Files

- **Based On:**
 - **Position:** The data in the specified position for the comparison.
 - **XPath:** The path to the XML field that is extracted.
 - **Use Selection:** Click to use the value of the current data selection for the extraction.
 - **Trim:** Select to trim empty characters at the beginning or the end of the field.
 - **Value:** A specified static text value.
 - **Value:** The text value to use in the comparison.
 - **Use selected text:** Uses the text in the current data selection as the **Value**. If multiple lines or elements are selected, only the first one is used.
 - **Field:** The contents of a specific field in the **Extracted Record**.
 - **Field:** The **Extracted Record** field to use in the comparison.
 - **JavaScript :** The result of a JavaScript **Expression**.
 - **Expression:** The JavaScript line that is evaluated. Note that the last value attribution to a variable is the one used as a result of the expression.
 - **Use JavaScript Editor:** Click to display the [Edit Script](#) dialog.
 - **Use selected text:** Inserts the text in the current data selection in the JavaScript **Expression**. If multiple lines or elements are selected, only the first one is used.
 - **Data Property:** The value of a data-level property set in the **Preprocessor** step.
 - **Record Property:** One of the local variables that you can create and that are reset for each document as opposed to data variables that are global because they are initialized only once at the beginning of each job.
 - **Automation Property:** The current value of a Document-level property set in the **Preprocessor** step.
 - **Extractor Property:** The value of an internal extractor variable:
 - **Counter:** The value of the current counter iteration in a **Repeat** step.
 - **Vertical Position:** The current vertical position on the page, either in Measure (PDF) or Line (Text and CSV).

- **Operators:**

- **is equal to:** The two specified value are identical for the condition to be **True**.
- **contains:** The first specified value contains the second one for the condition to be **True**.
- **is less than:** The first specified value is smaller, numerically, than the second value for the condition to be **True**.
- **is greater than:** The first specified value is larger, numerically, than the second value for the condition to be **True**.
- **is empty:** The first specified value is empty. With this operator, there is no second value.
- **Invert condition:** Inverts the result of the condition. For instance, **is empty** becomes **is not empty**.

Condition step properties

A **Condition** step is used when the data extraction must be based on specific criteria. See "Condition step" on page 145 for more information.

The properties of a Condition step become visible in the Step properties pane when the Condition step is selected on the Steps pane.

Description

This subsection is collapsed by default in the interface, to give more screen space to other important parts.

Name: The name of the step. This name will be displayed on top of the step's icon in the **Steps** pane.

Comments: The text entered here will be displayed in the tooltip that appears when hovering over the step in the **Steps** pane.

Rule tree

The **Rule tree** subsection displays the full combination of rules (defined below under **Condition**) as a tree, which gives an overview of how the conditions work together as well as the result for each of these conditions for the current record or iteration.

- To rename a rule, double click on its name from the **Rule tree** subsection.
- To change the way rules are combined, right-click "AND". Select OR or XOR instead. XOR means one or the other, but not both.

Condition

First, the **Condition List** displays the conditions in list form, instead of the tree form above. Three buttons are available next to the list:

- **Add condition:** Click to add a new rule. This will always branch the current condition as an "AND" operator.
- **Delete condition:** Delete the currently selected condition.

Conditions are made by comparison of two operands using a specific **Operator**.

Note

Both the **Left** and **Right** operands have the same properties.

- **Based On:**
 - **Position:** The data in the specified position for the comparison.
 - **Left** (Txt and PDF only): The start position for the data selection. Note that conditions are done on the current line, either the current cursor position, or the current line in a **Repeat** step.
 - **Right** (Txt and PDF only): The end position for the data selection.
 - **Height** (Txt and PDF only): The height of the selection box.
 - **Column** (CSV and Database only): Drop-down listing all fields in the Data Sample, of which the value will be used.
 - **XPath** (XML only): The path to the XML field that is extracted.
 - **Top offset:** The vertical offset from the current pointer location in the Data Sample (**Viewer**).
 - **Use Selection:** Click to use the value of the current data selection for the extraction.
 - **Trim:** Select to trim empty characters at the beginning or the end of the field.

- **Value:** A specified static text value.
 - **Value:** The text value to use in the comparison.
 - **Use selected text:** Uses the text in the current data selection as the **Value**. If multiple lines or elements are selected, only the first one is used.
- **Field:** The contents of a specific field in the **Extracted Record**.
 - **Field:** The **Extracted Record** field to use in the comparison.
- **JavaScript :** The result of a JavaScript **Expression**.
 - **Expression:** The JavaScript line that is evaluated. Note that the last value attribution to a variable is the one used as a result of the expression.
 - **Use JavaScript Editor:** Click to display the Edit Script dialog (see "Using scripts in the DataMapper" on page 255).
 - **Use selected text:** Inserts the text in the current data selection in the JavaScript **Expression**. If multiple lines or elements are selected, only the first one is used.
- **Data Property:** The value of a data-level property set in the Preprocessor (see "Preprocessor step" on page 140).
- **Record Property:** One of the local variables that you can create and that are reset for each document as opposed to data variables that are global because they are initialized only once at the beginning of each job.
- **Automation Property:** The current value of a Document-level property set in the Preprocessor step (see "Preprocessor step" on page 140).
- **Extractor Property:** The value of an internal extractor variable:
 - **Counter:** The value of the current counter iteration in a **Repeat** step.
 - **Vertical Position:** The current vertical position on the page, either in Measure (PDF) or Line (Text and CSV).
- **Operators:**
 - **is equal to:** The two specified value are identical for the condition to be **True**.
 - **contains:** The first specified value contains the second one for the condition to be **True**.
 - **is less than:** The first specified value is smaller, numerically, than the second value for the condition to be **True**.

- **is greater than**: The first specified value is larger, numerically, than the second value for the condition to be **True**.
- **is empty**: The first specified value is empty. With this operator, there is no second value.

Invert condition: Inverts the result of the condition. For instance, **is empty** becomes **is not empty**.

Multiple Conditions step properties

The **Multiple Conditions** step contains a number of Case conditions (one to start with) and a Default, to be executed when none of the other cases apply. Cases are executed from left to right. For more information see "Steps" on page 140.

The properties described below become visible in the Step properties pane when the Multiple Conditions step is selected in the Steps pane.

Description

This subsection is collapsed by default in the interface, to give more screen space to other important parts.

Name: The name of the step. This name will be displayed on top of the step's icon in the **Steps** pane.

Comments: The text entered here will be displayed in the tooltip that appears when hovering over the step in the **Steps** pane.

Condition

Left operand, Right operand

The **Left and right operand** can be **Based on**:

- **Position**: The data in the specified position for the comparison.
 - **Left** (Txt and PDF only): The start position for the data selection. Note that conditions are done on the current line, either the current cursor position, or the current line in a **Repeat** step.
 - **Right** (Txt and PDF only): The end position for the data selection.

- **Height** (Txt and PDF only): The height of the selection box.
- **Column** (CSV and Database only): Drop-down listing all fields in the Data Sample, of which the value will be used.
- **XPath** (XML only): The path to the XML field that is extracted.
- **Top offset**: The vertical offset from the current pointer location in the Data Sample (**Viewer**).
- **Use Selection**: Click to use the value of the current data selection for the extraction.
- **Trim**: Select to trim empty characters at the beginning or the end of the field.
- **Value**: A specified static text value.
 - **Value**: The text value to use in the comparison.
 - **Use selected text**: Uses the text in the current data selection as the **Value**. If multiple lines or elements are selected, only the first one is used.
- **Field**: The contents of a specific field in the **Extracted Record**.
 - **Field**: The **Extracted Record** field to use in the comparison.
- **JavaScript** : The result of a JavaScript **Expression**.
 - **Expression**: The JavaScript line that is evaluated. Note that the last value attribution to a variable is the one used as a result of the expression. See also: "DataMapper Scripts API" on page 252.
 - **Use JavaScript Editor**: Click to display the Edit Script dialog (see "Using scripts in the DataMapper" on page 255).
 - **Use selected text**: Inserts the text in the current data selection in the JavaScript **Expression**. If multiple lines or elements are selected, only the first one is used.
- **Data Property**: The value of a data-level property set in the Preprocessor step (see "Steps" on page 140).
- **Record Property**: One of the local variables that you can create and that are reset for each document as opposed to data variables that are global because they are initialized only once at the beginning of each job.
- **Automation Property**: The current value of a Document-level property set in the Preprocessor step (see "Steps" on page 140).

- **Extractor Property:** The value of an internal extractor variable:
 - **Counter:** The value of the current counter iteration in a **Repeat** step.
 - **Vertical Position:** The current vertical position on the page, either in Measure (PDF) or Line (Text and CSV).

Condition

The **Condition** drop-down displays the cases in list form. Three buttons are available next to the list:

- **Add case:** Click to add a new case to the step. It will be placed next to any existing cases.
- **Remove case:** Delete the currently selected case.
- **Order Cases:** Under the **Name** column, select a case, then click one of the buttons on the right (**Delete**, **Move Up**, **Move Down**) to delete or change the order of the cases in the list.

Operators

Case conditions are made by comparison of the two operands, left and right, using a specific **Operator**.

- **is equal to:** The two specified value are identical for the condition to be **True**.
- **contains:** The first specified value contains the second one for the condition to be **True**.
- **is less than:** The first specified value is smaller, numerically, than the second value for the condition to be **True**.
- **is greater than:** The first specified value is larger, numerically, than the second value for the condition to be **True**.
- **is empty:** The first specified value is empty. With this operator, there is no second value.
- **Invert condition:** Inverts the result of the condition. For instance, **is empty** becomes **is not empty**.

Goto step properties

The **Goto** step moves the pointer within the source data to a position that is relative to the top of the record or to the current position. See also: "Steps" on page 140.

The properties of the Goto step described in this topic become visible in the Step properties pane when you select the Goto step on the Steps pane.

Description

This subsection is collapsed by default in the interface, to give more screen space to other important parts.

Name: The name of the step. This name will be displayed on top of the step's icon in the **Steps** pane.

Comments: The text entered here will be displayed in the tooltip that appears when hovering over the step in the **Steps** pane.

Goto Definition

With each type of source data, the movement of the cursor is defined in a specific way.

Text file

- **Target Type:** Defines the type of jump .
 - **Line:** Jumps a certain number of lines or to a specific line.
 - **From:** Defines where the jump begins:
 - **Current Position:** The **Goto** begins at the current cursor position.
 - **Top of record:** The **Goto** begins at line 1 of the source record.
 - **Move by:** Enter the number of lines or pages to jump.
 - **Page:** Jumps between pages or to a specific page.
 - **From:** Defines where the jump begins:
 - **Current Position:** The **Goto** begins at the current cursor position.
 - **Top of record:** The **Goto** begins at line 1 of the source record.
 - **Move by:** Enter the number of lines or pages to jump.
 - **Next line with content:** Jumps to the next line that has contents, either anywhere on the line or in specific columns.
 - **Inspect entire page width:** When checked, the **Next line with content** and **Next occurrence of** options will look anywhere on the line. If unchecked,

options appear below to specify in which area of each line the **Gotostep** checks in:

- **Left:** The starting column, inclusively.
 - **Right:** The end column, inclusively.
 - **Use selection:** Click while a selection is made in the Data Viewer to automatically set the left and right values to the left and right edges of the selection.
- **Next occurrence of:** Jumps to the next occurrence of specific text or a text pattern, either anywhere on the line or in specific columns.
 - **Inspect entire page width:** When checked, the **Next line with content** and **Next occurrence of** options will look anywhere on the line. If unchecked, options appear below to specify in which area of each line the **Gotostep** checks in:
 - **Left:** The starting column, inclusively.
 - **Right:** The end column, inclusively.
 - **Use selection:** Click while a selection is made in the Data Viewer to automatically set the left and right values to the left and right edges of the selection.
 - **Expression:** Enter the text or **Regex** expression to look for on the page.
 - **Use selection:** Click while a selection is made in the Data Viewer to copy the contents of the first line of the selection into the **Expression** box.
 - **Use regular expression:** Check so that the **Expression** box is treated as a regular expression instead of static text. For more information on using **Regular Expressions (regex)**, see the [Regular-Expressions.info Tutorial](http://Regular-Expressions.info).

PDF File

- **Target Type:** Defines the type of jump .
 - **Physical distance:**
 - **From:** Defines where the jump begins:
 - **Current Position:** The **Goto** begins at the current cursor position.
 - **Top of record:** The **Goto** begins at line 1 of the source record.
 - **Move by:** Enter distance to jump.

- **Page:** Jumps between pages or to a specific page.
 - **From:** Defines where the jump begins:
 - **Current Position:** The **Goto** begins at the current cursor position.
 - **Top of record:** The **Goto** begins at line 1 of the source record.
 - **Move by:** Enter the number pages to jump.
- **Next line with content:** Jumps to the next line that has contents, either anywhere on the line or in specific columns.
 - **Inspect entire page width:** When checked, the **Next line with content** and **Next occurrence of** options will look anywhere on the line. If unchecked, options appear below to specify in which area of each line the **Gotostep** checks in:
 - **Left:** The starting column, inclusively.
 - **Right:** The end column, inclusively.
 - **Use selection:** Click while a selection is made in the Data Viewer to automatically set the left and right values to the left and right edges of the selection.
- **Next occurrence of:** Jumps to the next occurrence of specific text or a text pattern, either anywhere on the line or in specific columns.
 - **Inspect entire page width:** When checked, the **Next line with content** and **Next occurrence of** options will look anywhere on the line. If unchecked, options appear below to specify in which area of each line the **Gotostep** checks in:
 - **Left:** The starting column, inclusively.
 - **Right:** The end column, inclusively.
 - **Use selection:** Click while a selection is made in the Data Viewer to automatically set the left and right values to the left and right edges of the selection.
 - **Expression:** Enter the text or **Regex** expression to look for on the page.
 - **Use selection:** Click while a selection is made in the Data Viewer to copy the contents of the first line of the selection into the **Expression** box.

- **Use regular expression:** Check so that the **Expression** box is treated as a regular expression instead of static text. For more information on using **Regular Expressions (regex)**, see the [Regular-Expressions.info Tutorial](#).

CSV File

- **From (CSV files):** Defines where the jump begins:
 - **Current Position:** The **Goto** begins at the current cursor position.
 - **Move by:** Enter the number of lines or pages to jump.
 - **Top of record:** The **Goto** begins at line 1 of the source record.
 - **Move to:** Enter the number of lines or pages to jump.

XML File

- **Destination (XML files):** Defines what type of jump to make:
 - **Sibling element:** Jumps the number of siblings (nodes at the same level) defined in the **Move by** option.
 - **Sibling element with same name:** Jumps the number of same name siblings (nodes at the same level of which the node is the same name) defined in the **Move by** option.
 - **Element, from top of record:** Jumps to the specified node. The XPATH in the **Absolute XPATH** option starts from the root node defined by */*.
 - **Element from current position:** Jumps to a position relative to the current position of the cursor. The XPATH in the **Relative XPATH** option defines where to go, *../* goes up a level, */* refers to the current level.
 - **Level Up/Down:** Jumps up or down one node level (up to the parent, down to a child). The number of levels to change is defined in the **Move by** option.

Postprocessor step properties

The Postprocessor step does not run for every Source Record in the Data Sample. It runs once, at the end of the Steps, after all records have been processed. For more information see "Postprocessor step" on page 150.

The properties described below become visible in the Step properties pane when the Postprocessor step is selected in the Steps pane.

Description

This subsection is collapsed by default in the interface, to give more screen space to other important parts.

Name: The name of the step. This name will be displayed on top of the step's icon in the **Steps** pane.

Comments: The text entered here will be displayed in the tooltip that appears when hovering over the step in the **Steps** pane.

Postprocessor

The Postprocessor subsection defines what postprocessors run on the Data Sample at the end of the data mapping workflow. Each Postprocessor runs in turn, using the result of the previous one as an input.

- **Name:** The name to identify the Postprocessor.
- **Type:** The type of Postprocessor. Currently there is a single type available.
 - **JavaScript :** Runs a JavaScript Expression to modify the Data Sample. See "DataMapper Scripts API" on page 252.
 - **Use JavaScript Editor:** Click to display the Edit Script dialog (see "Using scripts in the DataMapper" on page 255).
- **Add Postprocessor:** Click to add a new Postprocessor. Its settings can be modified once it is added.
- **Remove Postprocessor:** Click to remove the currently selected Postprocessor.
- **Move Up:** Click to move the Postprocessor up one position.
- **Move Down:** Click to move the Postprocessor down one position.
- **Export:** Click to export the current Postprocessor configuration and content to a file.
- **Import:** Click to import a Postprocessor configuration and content from an external file.




JavaScript

- **Expression:** The JavaScript expression that will run on the Data Sample. See "DataMapper Scripts API" on page 252.
- **Use JavaScript Editor:** Click to display the Script Editor dialog.
- **Use selected text:** Uses the text in the current data selection as the Value. If multiple lines or elements are selected, only the first one is used.

Toolbar

In the DataMapper module, the following buttons are available in the top toolbar.



File manipulation













- **New** : Displays the **New** wizard where a new data mapping configuration or a new template can be created.
- **Open** : Displays the **Open** dialog to open an existing data mapping configuration.
- **Save** : Saves the current data mapping configuration. If the configuration has never been saved, the **Save As...** dialog is displayed.

Step manipulation

Note

All steps except the Action step require an active data selection in the Data Viewer (see "Selecting data" on page 122 and "The Data Viewer" on page 200).

- **Add Extract Step** : Adds an Extract Step with one or more extract fields. If more than one line or field is selected in the Data Viewer, each line or field will have an extract field.
- **Add Goto Step** : Adds a Goto step that moves the selection pointer to the beginning of the data selection. For instance if an XML node is selected, the pointer moves to where that node is located.

- **Add Condition Step**  : Adds a condition based on the current data selection. The "True" branch gets run when the text is found on the page. Other conditions are available in the step properties once it has been added.
- **Add Repeat Step**  : Adds a loop that is based on the current data selection, and depending on the type of data. XML data will loop on the currently selected node, CSV loops for all rows in the record. In Text and PDF data, if the data selection is on the same line as the cursor position, the loop will be for each line until the end of the record. If the data selection is on a lower line, the loop will be for each line until the text in the data selection is found at the specified position on the line (e.g. until "TOTAL" is found).
- **Add Extract Field**  : Adds the data selection to the selected **Extract** step, if an extract step is currently selected. If multiple lines, nodes or fields are selected, multiple extract fields are added simultaneously.
- **Add Multiple Conditions**  : Adds a condition that splits into multiple case conditions.
- **Add Action Step**  : Adds a step to create a custom JavaScript snippet. See "DataMapper Scripts API" on page 252 for more details.
- **Cut Step**  : Removes the currently selected step and places it in the clipboard. If the step is a Repeat or a Condition, all steps under it are also placed in the clipboard. If there is already a step in the clipboard, it will be overwritten.
- **Copy Step**  : Places a copy of the currently selected step in the clipboard. The same details as the Cut step applies.
- **Paste Step**  : Takes the step or steps in the clipboard and places them after the currently selected step.
- **Delete Step**  : Deletes the currently selected step. If the step is a Repeat or Condition, all steps under it are also deleted.
- **Ignore Step**  : Click to set the step to be ignored (aka disabled). Disabled steps do not run when in DataMapper and do not execute when the data mapping configuration is executed in Workflow. However, they can still be modified normally.
- **Validate All Records**  : Runs the process on all records and verifies that no errors are present in any of the records. Errors are displayed in the Messages pane ("Messages pane" on page 202).
- **Add Data Sample**  : Displays a dialog to open a new Data Source to add it as a Data Sample in the data mapping configuration. Data Samples are visible in the Settings pane ("Settings pane" on page 203).

Welcome Screen

The **Welcome Screen** appears when first starting up PlanetPress Connect. It offers some useful shortcuts to resources and to recent documents and data mapping configurations.

If you are new to PlanetPress Connect and you don't know where to start, see "Welcome to PlanetPress Connect 1.8" on page 14.

The Welcome Screen can be brought back in two ways:

- The **Welcome Screen** button in the "Toolbars" on page 771.
- From the Menus in **Help, Welcome Screen**.

Contents

- **Activation:** Click to open the **Objectif Lune Web Activation Manager**.
- **Release Notes:** Opens the current **Release Notes** for PlanetPress Connect.
- **Website:** Opens the PlanetPress Connect website.
- **Take A Tour:** Click to open the YouTube Playlist giving you a tour of the software.
- **Use the DataMapper to...:**
 - **Create a New Configuration:** Opens the [Creating a New Configuration](#) screen.
 - **Open an Existing Configuration:** Click to open the standard **Browse** dialog to open an existing data mapping configuration.
 - **Recent Configurations:** Lists recently used configurations. Click any configuration to open it in the DataMapper module.
- **Use the Designer to...:**
 - **Create a New Template:** Lets you choose a Context to create a new template without a Wizard.
 - **Browse Template Wizards:** Displays a list of available Template Wizards, producing premade templates with existing demo content; see "Creating a template" on page 304.
 - **Open an Existing Template:** Click to open the standard **Browse** dialog to open an existing template.

- **Recent Templates:** Lists recently used templates. Click any template to open it in the Designer module.
- **Other Resources:**
 - **Documentation:** Opens this documentation.
 - **Courses (OL Learn):** Opens the [Objectif Lune e-Learning Center](#).
 - **User Forums:** Opens the [Questions & Answer](#) forums.

DataMapper Scripts API

This page describes the different features available in scripts created inside DataMapper. See "Using scripts in the DataMapper" on page 255.

Objects

Name	Description	Available in scripts of type
"Objects" on page 262	A ScriptableAutomation object encapsulating the properties of the PlanetPress Workflow process that triggered the current operation.	Boundaries, all steps except Goto
"boundaries" on page 264	An object encapsulating properties and methods allowing to define the boundaries of each document in the job.	Boundaries
"data" on page 269	A data object encapsulating properties and methods pertaining to the original data stream.	Boundaries, all steps except Goto
"db" on page 282	An object that allows to connect to a database.	Boundaries, all steps except Goto

Name	Description	Available in scripts of type
"logger" on page 283	An object that allows to log error, warning or informational messages.	Boundaries, all steps except Goto
"record" on page 283	The current record in the main data set.	Extract, Condition, Repeat and Multiple Conditions steps
"region" on page 284	An object that defines a subsection of the input data.	Boundaries
"sourceRecord" on page 286	An object containing properties specific to the current source record being processed.	Boundaries, all steps except Goto and Postprocessor
"steps" on page 287	An object encapsulating properties and methods pertaining to the current data mapping configuration.	Extract, Condition, Repeat and Multiple Conditions steps

Functions

These functions are available in Boudaries and Steps scripts.

Name	Description
copyFile()	Copies a file to the target file path, replacing it if it already exists.
"createHttpRequest()" on page 294	Creates a new HTTP Request Object.
createTmpFile()	Creates a file with a unique name in the temporary work folder and returns a file object.
deleteFile()	Deletes a file.
execute()	Calls an external program and waits for it to end.
newByteArray()	Returns a new byte array.
newCharArray()	Returns a character array.
newDoubleArray()	Returns a double array.
newFloatArray()	Returns a float array.
newIntArray()	Returns an integer array.
newLongArray()	Returns a long array.
newStringArray()	Returns a string array.
openBinaryReader()	Opens a file as a binary file for reading purposes.
openBinaryWriter()	Opens a file as a binary file for writing purposes.
openTextReader()	Opens a file as a text file for reading purposes.
openTextWriter()	Opens a file as a text file for writing purposes.

Using scripts in the DataMapper

In the DataMapper every part of the extraction process can be customized using scripts.

A script can be used to set **boundaries** for a data source (see "Setting boundaries using JavaScript" on page 257). The script determines where a new record starts.

Scripts can also be used in different steps in the extraction workflow. You can:

- Modify the incoming data prior to executing the rest of the extraction workflow, via a **Preprocessor** (see "Preprocessor step" on page 140).
- Edit extracted data in a field of the Data Model using a **Post function** script (entered on the Step properties pane, under Field Definition; see "Modifying extracted data" on page 159 and "Settings for location-based fields in a Text file" on page 221).
- Enter a script in a **JavaScript-based field** (see "JavaScript-based field" on page 156). Note that the last value attribution to a variable is the one used as the result of the expression.
It is possible to refer to previously extracted fields if they are extracted higher in this list or in previous **Extract** steps in the extraction workflow.
- Let an **Action** step run a JavaScript, or use JavaScript to add a value to a property defined in the Preprocessor step.
- Change the left and right operands in a **Condition** step to a JavaScript expression. (On the Step properties pane, under Condition, set Based on to Javascript; see "Condition step properties" on page 238 and "Left operand, Right operand" on page 241.)
- Further process the resulting record set after the entire extraction workflow has been executed, via a **Postprocessor** (see "Postprocessor step" on page 150).

The script can always be written directly in a small script area or in the **Edit script** dialog. To invoke this dialog click the **Use JavaScript Editor** button .

Tip

In the **Edit script** dialog, press **Ctrl + Space** to bring up the list of available JavaScript objects and functions (see [Datamapper API](#)). Use the arrow keys to select a function or object and press enter to insert it. Type a dot after the name of the function or object to see which features are

subsequently available.

Keyboard shortcuts for the script editor are listed in the following topic: "Keyboard shortcuts" on page 738.

Syntax rules

In the DataMapper, all scripts must be written in **JavaScript**, following JavaScript syntax rules. For example, each statement should end with ; and the keywords that can be used, such as **var** to declare a variable, are JavaScript keywords. There are countless tutorials available on the Internet to familiarize yourself with the JavaScript syntax. For a simple script all that you need to know can be found on the following web pages: http://www.w3schools.com/js/js_syntax.asp and http://www.w3schools.com/js/js_if_else.asp. A complete JavaScript guide for beginners can be found here: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>.

DataMapper API

Certain features that can be used in a DataMapper script do not exist in the native JavaScript library. These are additional JavaScript features, designed for use in Connect only. All features designed for use in the DataMapper are listed in the DataMapper API (see [DataMapper API](#)).

External JavaScript libraries

The External JS Libraries box on the Settings pane lets you add JavaScript libraries to your configuration and displays all the libraries that have been imported (see "Settings pane" on page 203).

You can use JavaScript libraries to add more JavaScript functionality to your data mapping configuration. Any functions included in a JavaScript library that is imported in a data mapping configuration will be available in Preprocessor scripts as well as Action tasks, Post functions and JavaScript-based extraction steps.

Take the following JavaScript function, for example:

```
function myAddFunction(p1, p2) {  
    return p1 + p2;  
};
```

If this is saved as myFunction.js and imported, then the following would work anywhere in the configuration:

```
var result = myAddFunction(25, 12); // returns 37!
```

Setting boundaries using JavaScript

As soon as you select the **On Script** option as the trigger for establishing record boundaries (see "Record boundaries" on page 117), you are instructing the DataMapper to read the source file sequentially and to trigger an event each and every time it hits a delimiter. (What a delimiter is, depends on the source data and the settings for that data; see "Input data settings (Delimiters)" on page 115).

In other words, the script will be executed - by default - as many times as there are delimiters in the input data.

If you know, for instance, that a PDF file only contains documents that are 3 pages long, your script could keep count of the number of times it's been called since the last boundary was set (that is, the count of delimiters that have been encountered). Each time the count is a multiple of 3, it could set a new record boundary. This is basically what happens when setting the trigger to **On Page** and specifying 3 as the Number of Pages.

Note

Remember that a boundary script is being called on each new delimiter encountered by the DataMapper parsing algorithm. If for instance a database query returns a million records, the script will be executing a million times! Craft your script in such a way that it doesn't waste time examining all possible conditions. Instead, it should terminate as soon as any condition it is evaluating is false.

Accessing data

Data available inside each event

Every time a delimiter is encountered, an event is triggered and the script is executed. The event gives the script access to the data between the current location - the start of a row, line or page - and the next delimiter. So at the beginning of the process for a PDF or text file, you have access to the first page only, and for a CSV or for tabular data, that would be the first row or record.

This means that you can:

- Examine the data found in between delimiters for specific conditions.
- Examine specific regions of that data, or the available data as a whole.
- Compare the contents of one region with another.
- Etc.

To access this data in the script, use the **get()** function of the **boundaries** object. This function expects different parameters depending on the type of source file; see "Example" on page 266.

Getting access to other data

Data that is not passed with the event, but that is necessary to define the record boundaries, can be stored in the **boundaries** object using the **setVariable** function (see "boundaries" on page 264 and "Example" on page 268). The data can be retrieved using the boundaries' **getVariable** function (see "getVariable()" on page 266).

This way the script can access values that were evaluated in previous pages or rows, across delimiters, so you can easily set record boundaries that span over multiple delimiters.

For more information on the syntax, please refer to "DataMapper Scripts API" on page 252.

Examples

Basic example using a CSV file

Imagine you are a classic rock fan and you want to extract the data from a CSV listing of all the albums in your collection. Your goal is to extract records that change whenever the artist OR the release year changes.

Here's what the CSV looks like:

```
"Artist","Album","Released"
"Beatles","Abbey Road",1969
"Beatles","Yellow Submarine",1969
"Led Zeppelin","Led Zeppelin 1",1969
"Led Zeppelin","Led Zeppelin 2",1969
"Beatles","Let it be",1969
"Rolling Stones","Let it bleed",1969
"Led Zeppelin","Led Zeppelin 3",1970
"Led Zeppelin","Led Zeppelin 4",1971
"Rolling Stones","Sticky Fingers",1971
```

Note

The first line is just the header with the names of the CSV columns. The data is already sorted per year, per artist, and per album.

Your goal is to examine two values in each CSV record and to act when either changes. The DataMapper GUI allows you to specify a **On Change** trigger, but you can only specify a single field. So for instance, if you were to set the record boundary when the "Released" field changes, you'd get the first four lines together inside a single record. That's not what you want since that would include albums from several different artists. And if you were to set it when the "Artist" field changes, the first few records would be OK but near the end, you'd get both the Led Zeppelin 3 and led Zeppelin 4 albums inside the same record, even though they were released in different years.

Essentially, we need to combine both these conditions and set the record boundary when EITHER the year OR the artist changes.

Here's what the script would look like:

```
/* Read the values of both columns we want to check */
var zeBand = boundaries.get(region.createRegion("Artist"));
var zeYear = boundaries.get(region.createRegion("Released"));

/* Check that at least one of our variables holding previous values
has been initialized already, before attempting to compare the
values */

if (boundaries.getVariable("lastBand")!=null) {
    if (zeBand[0] != boundaries.getVariable("lastBand") || zeYear[0]
!= boundaries.getVariable("lastYear") )
    {
        boundaries.set();
    }
}
boundaries.setVariable("lastBand", zeBand[0]);
boundaries.setVariable("lastYear", zeYear[0]);
```

- The script first reads the two values from the input data, using the createRegion() method (see: "Example" on page 285). For a CSV/database data type, the parameter it expects is

simply the column name. The region is passed as a parameter to the `get()` method, which reads its contents and converts it into an array of strings (because any region, even a CSV field, may contain several lines).

- To "remember" the values that were processed the last time the event was triggered, we use variables that remain available in between events. Note that these variables are specific to the Boundary context and not available in any other scripting context in the `DataMapper`.
- The script first checks if those values were initialized. If they weren't, it means this is the first iteration so there's no need to compare the current values with previous values since there have been none yet. But if they have already been initialized, then a condition checks if either field has changed since last time. If that's the case, then a boundary is created through the `set()` method.
- Finally, the script stores the values it just read in the variables using the `setVariables()` method. They will therefore become the "last values encountered" until the next event gets fired. When called, `setVariables()` creates the specified variable if it doesn't already exist and then sets the value to the second parameter passed to the function.

You can try it yourself. Paste the data into the text editor of your choice and save the file to `Albums.csv`. Then create a new `DataMapper` configuration and load this CSV as your data file. In the Data Input Settings, make sure you specify the first row contains field names and set the **Trigger** to **On script**. Then paste the above JavaScript code in the **Expression** field and click the **Apply** button to see the result.

Basic example using a text file

This example is similar to the previous example, but now the data source is a plain text file that looks like this:

Beatles	Abbey Road	1969
Beatles	Yellow Submarine	1968
Led Zeppelin	Led Zeppelin 1	1969
Led Zeppelin	Led Zeppelin 2	1969
Beatles	Let it be	1970
Rolling Stones	Let it bleed	1969
Led Zeppelin	Led Zeppelin 3	1970
Led Zeppelin	Led Zeppelin 4	1971
Rolling Stones	Sticky Fingers	1971

The purpose of the script, again, is to set the record boundary when EITHER the year OR the artist changes.

The script would look like this:

```
/* Read the values of both columns we want to check */
var zeBand = boundaries.get(region.createRegion(1,1,30,1));
var zeYear = boundaries.get(region.createRegion(61,1,65,1));

/* Check that at least one of our variables holding previous values
have been initialized already, before attempting to compare the
values */

if (boundaries.getVariable("lastBand")!=null) {
    (zeBand[0]!=boundaries.getVariable("lastBand") || zeYear
[0]!=boundaries.getVariable("lastYear"))
    {
        boundaries.set();
    }
}
boundaries.setVariable("lastBand",zeBand[0]);
boundaries.setVariable("lastYear",zeYear[0]);
```

This script uses the exact same code as used for CSV files, with the exception of parameters expected by the createRegion() method. The get method adapts to the context (the data source file) and therefore expects different parameters to be passed in order to achieve the same thing. Since a text file does not contain column names as a CSV does, the API expects the text regions to be defined using physical coordinates. In this instance: Left, Top, Right, Bottom.

To try this code, paste the data into a text editor and save the file to Albums.txt. Then create a new DataMapper configuration and load this Text file as your data file. In the Data Input Settings, specify **On lines** as the **Page delimiter** type with the number of lines set to 1. When you now set the boundary **Trigger** to **On script**, the file will be processed line per line (triggering the event on each line). Paste the above code in the JavaScript expression field and click the **Apply** button to see the result.

Note

The PDF context also expects physical coordinates, just like the Text context does, but since PDF

pages do not have a grid concept of lines and columns, the above parameters would instead be specified in millimeters relative to the upper left corner of each page. So for instance, to create a region for the Year, the code might look like this:

```
region.createRegion(190,20,210,25)
```

which would create a region located near the upper right corner of the page.

That's the only similarity, though, since the script for a PDF would have to look through the entire page and probably make multiple extractions on each one since it isn't dealing with single lines like the TXT example given here.

For more information on the API syntax, please refer to "DataMapper Scripts API" on page 252.

Objects

automation

Returns a **ScriptableAutomation** object encapsulating the properties of the PlanetPress Workflow process that triggered the current operation.

Note

The `automation` object available in Designer scripts is not of the same type. It has different properties.

Properties

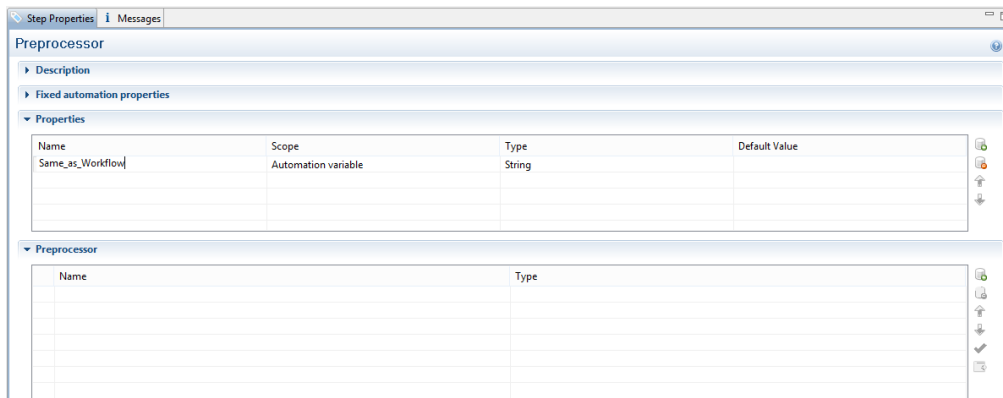
The following table lists the properties of the **automation** object. These are available in Boundaries scripts, with all file types.

Property	Description
jobInfo	Returns a ScriptableAutomation object containing JobInfo 1 to 9 values from PlanetPress Workflow.

Property	Description
properties	Returns a ScriptableAutomation object containing additional information (file name, process name and task ID) from PlanetPress Workflow.
variables	Returns a ScriptableAutomation object containing the list of local and global variables defined by the user in PlanetPress Workflow. Note that there is no way to distinguish local variables from global ones (local variables take precedence over global variables). To be used in the DataMapper, variables must have already been defined in the Preprocessor step as Automation variables. The Preprocessor step attempts to match variable names passed by the Workflow process to those defined inside the step.

Accessing automation properties

To make a Workflow **variable** accessible in scripts, it must first be declared in the Properties of the Preprocessor step (see "Properties" on page 217). Both the name and type of the variable must be the same as the variable in Workflow.



The other properties are accessible as they are.

Examples

To access JobInfo 1 to 9 from Workflow:

```
automation.jobInfo.JobInfo1;
```

To access ProcessName, OriginalFilename or TaskIndex from Workflow:

automation.properties.OriginalFilename;

To access Workflow variables (declared in the Preprocessor properties):

automation.variables.Same_as_workflow;

boundaries

Returns a **boundaries** object encapsulating properties and methods allowing to define the boundaries of each document in the job.

This object is available when triggering document boundaries **On script**.

Properties

The following table lists the properties of the boundaries object.

Property	Return Type
currentDelim	A read-only 1-based index (number) of the current delimiter in the file. In other words, the Beginning Of File (BOF) delimiter equals 1. It indicates the position of the current delimiter relative to the last document boundary

Methods

The following table describes the functions of the boundaries object. They are available with all file types.

Method	Description	Script type
"find()" on the next page	Finds the first occurrence of a string starting from the current position.	Boundaries Preprocessor, Extract, Condition, Repeat, Action, and Postprocessor steps
get()	Retrieves an array of strings.	Boundaries

Method	Description	Script type
getVariable()	Retrieves a value of a variable stored in the <code>boundaries</code> object.	Boundaries
set()	Sets a new record boundary. (See: "Record boundaries" on page 117.)	Boundaries
setVariable()	Sets a <code>boundaries</code> variable to the specified value, automatically creating the variable if it doesn't exist yet.	Boundaries

find()

Method of the `boundaries` object that finds a string in a region of the data source file. The method returns a smaller region which points to the exact location where the match was found.

`find(stringToFind, in_Region)`

Finds the string `stringToFind` in a rectangular region defined by `inRegion`.

stringToFind

String to find.

in_Region

The `inRegion` can be created prior to the call to `find()` with the `region.createRegion()` method. It depends on the type of data source how a region is defined; see "Example" on page 285.

The `find()` method returns a different `region` object whose `range` property is adjusted to point to the exact physical location where the match was found. This will always be a subset of the `in_Region.range` property. It can be used to determine the exact location where the match occurred.

Use `boundaries.get()` to retrieve the actual text from the resulting region; see "Example" on the facing page.

get()

The `get()` method reads the contents of a `region` object and converts it into an array of strings (because any region may contain several lines).

How the `region` is defined, depends on the type of source data; see "region" on page 284 and "Example" on page 285.

get(in Region)

in_Region

A region object. What type of object this is depends on the type of source data, however in any case the `region` object can be created with a call to `region.createRegion()`; see "Example" on page 285.

Example

This script retrieves all text from the `Email_Address` field in a CSV or database file.

```
boundaries.get(region.createRegion("Email_Address"));
```

getVariable()

Method that retrieves the value currently stored in a variable.

Note

Boundary variables are carried over from one iteration of the Boundaries script to the next, while native JavaScript variables are not.

getVariable(varName)

varName

String name of the variable from which the value is to be retrieved. If the variable does not exist, the value `null` is returned. It is considered good practice (almost mandatory, even) to always check whether a variable is defined before attempting to access its value.

set()

Sets a new DataMapper record boundary.

set(delimiters)

delimiters

*Sets a new record boundary. The **delimiters** parameter is an offset from the current delimiter, expressed in an integer that represents a number of delimiters.*

If this parameter is not specified, then a value of 0 is assumed. A value of 0 indicates the record boundary occurs on the current delimiter.

A negative value of -n indicates that the record boundary occurred -n delimiters before the current delimiter.

A positive value of n indicates that the record boundary occurs +n delimiters after the current delimiter.

Note

Specifying a positive value not only sets the DataMapper record boundary but it also advances the current delimiter to the specified delimiter. That's where the processing resumes. This allows you to skip some pages/records when you know they do not need to be examined. Negative (or 0) values simply set the boundary without changing the current location.

Example

This script sets a boundary when the text TOTAL is found on the current page in a PDF file. The number of delimiters is set to 1, so the boundary is set on the next delimiter, which is the start of the next page.

```
if (boundaries.find("TOTAL", region.createRegion
(10,10,215,279)).found) {
    boundaries.set(1);
}
```

Assume you want to set record boundaries whenever the text "TOTAL" appears in a specific region of the page of a PDF file, but the PDF file has already been padded with blank pages for duplexing purposes. The boundary should therefore be placed at the end of the page where the match is found if that match occurs on an even page, or at the end of the next blank page, if the match occurs on an odd page. Recall that for PDF files, the natural delimiter is a PDF page.

The JavaScript code would look something like the following:

```
var myRegion = region.createRegion(150,220,200,240);
if(boundaries.find("TOTAL", myRegion).found) {
    /* a match was found. Check if we are on a odd or even page and
```

```

set the Boundary accordingly */
    if((boundaries.currentDelim % 2) !=0 ) {
        /* Total is on odd page, let's set the document Boundary on
delimiter further, thereby skipping the next blank page */
        boundaries.set(1);
    } else {
        /* Total is on an even page, set the document Boundary to the
current delimiter */
        boundaries.set();
    }
}
}

```

setVariable()

This method sets a variable in the `boundaries` to the specified value, automatically creating the variable if it doesn't exist yet.

Note

Boundary variables are carried over from one iteration of the Boundaries script to the next, while native JavaScript variables are not.

setVariable(varName, varValue)

Sets variable **varName** to value **varValue**.

varName

String name of the variable of which the value is to be set.

varValue

Object; value to which the variable has to be set.

Example

This script examines a specific region and stores its contents in a variable in the boundaries.

```

var addressRegion = region.createRegion(10, 30, 100, 50);
var addressLines = boundaries.get(addressRegion);
boundaries.setVariable("previousLines",addressLines);

```


data

Returns a data object encapsulating properties and methods pertaining to the original data stream.

Properties

The following table lists the properties of the data object.

Property	Description	Return type
filename	The path of the input file.	Returns the fully qualified file name of the temporary work file being processed.
properties	Contains properties declared in the preprocessor step (see Preprocessor Step Properties for details).	Returns an array of properties defined in the Preprocessor step with the data scope (i.e. statically set at the start of the job).

Methods

The following table lists the methods of the data object.

Method	Description	Script type	File type
"Examples" on page 271	Extracts the text value from a rectangular region.	Extract, Condition, Repeat, and Action steps	All
"extractMeta ()" on page 277	Extracts the value of a metadata field.	Extract, Condition, Repeat, and Action steps	All
"fieldExists ()" on page 277	Method that returns true if the specified metadata field, column or node exists.	Boundaries Preprocessor, Extract, Condition, Repeat, Action,	All

Method	Description	Script type	File type
		and Postprocessor steps	
"Examples" on page 279	Finds the first occurrence of a string starting from the current position.	Boundaries Preprocessor, Extract, Condition, Repeat, Action, and Postprocessor steps	All
"Examples" on page 282	Finds the first match for a regular expression pattern starting from the current position.	Extract, Condition, Repeat, Multiple Conditions and Action steps	Text, PDF

extract()

Extracts the text value from selected data: a node path, column, or rectangular region, depending on the type of data source.

This method always returns a String.

extract(left, right, verticalOffset, regionHeight, separator)

Extracts a value from a position in a **text** file. Coordinates are expressed as characters (horizontally) or lines (vertically).

left

Number that represents the distance, measured in characters, from the left edge of the page to the left edge of the rectangular region. The leftmost character is character 1.

right

Number that represents the distance, measured in characters, from the left edge of the page to the right edge of the rectangular region.

verticalOffset

Number that represents the current vertical position, measured in lines.

regionHeight

Number that represents the total height of the region, measured in lines.

Setting the regionHeight to 0 instructs the DataMapper to extract all lines starting from the given position until the end of the record.

Specifying an extraction height that is longer than the number of remaining lines results in a "step out of bound" error message.

separator

String inserted between all lines returned from the region. If you don't want anything to be inserted between the lines, specify an empty string ("").

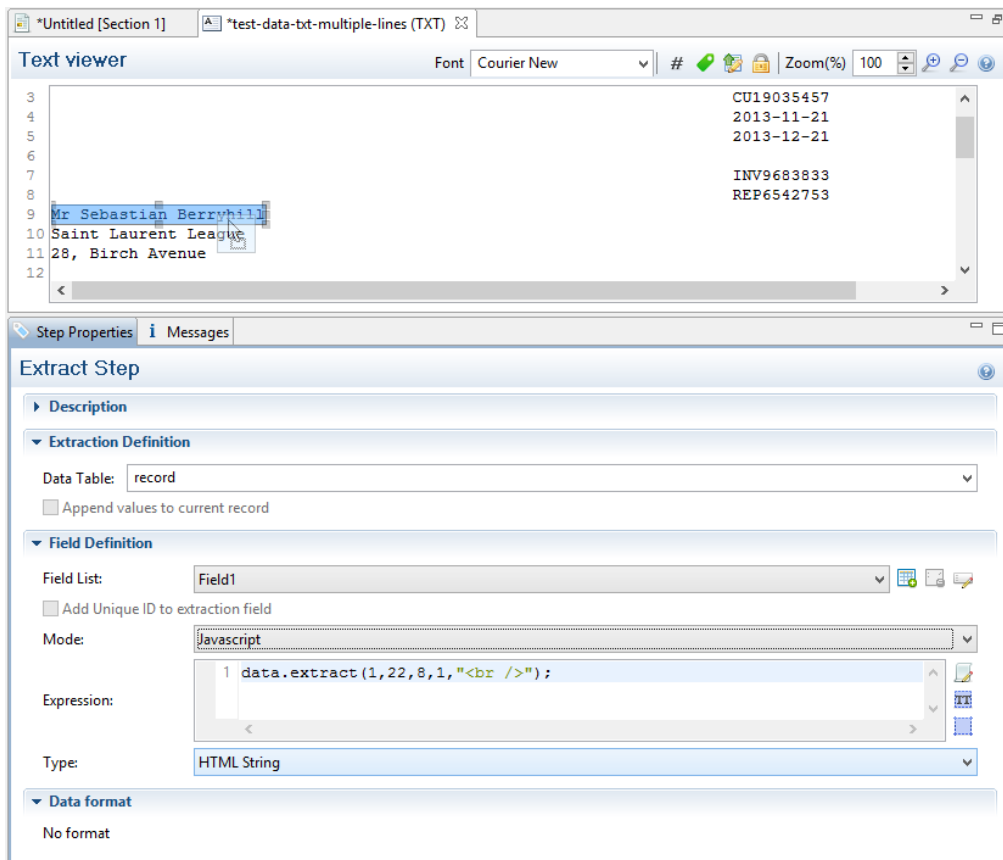
Tip

- "
" is a very handy string to use as a separator. When the extracted data is inserted in a Designer template, "
" will be interpreted as a line break, because
 is a line break in HTML and Designer templates are actually HTML files.
- Setting the regionHeight to 0 makes it possible to extract a variable number of lines at the end of a record.

Examples

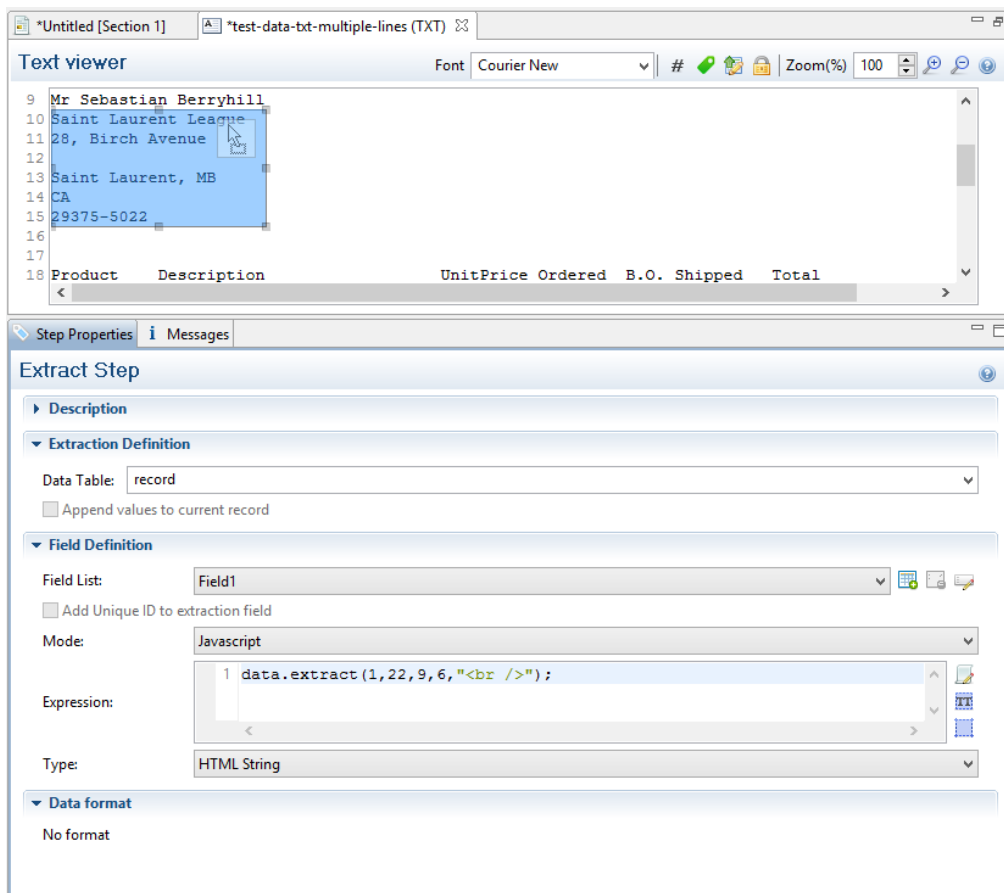
Example 1:

The script command `data.extract(1,22,8,1,"
");` means that the left position of the extracted information is located at character 1, the right position at character 22, the offset position is 8 (since the line number is 9) and the regionHeight is 1 (to select only 1 line). Finally, the "
" string is used for concatenation.



Example 2:

The script command **data.extract(1,22,9,6,"
");** means that the left position of the extracted information is located at 1, the right position at 22, the offset position is 9 (since the first line number is 10) and the regionHeight is 6 (6 lines are selected). Finally, the "
" string is used for concatenation.



extract(xPath)

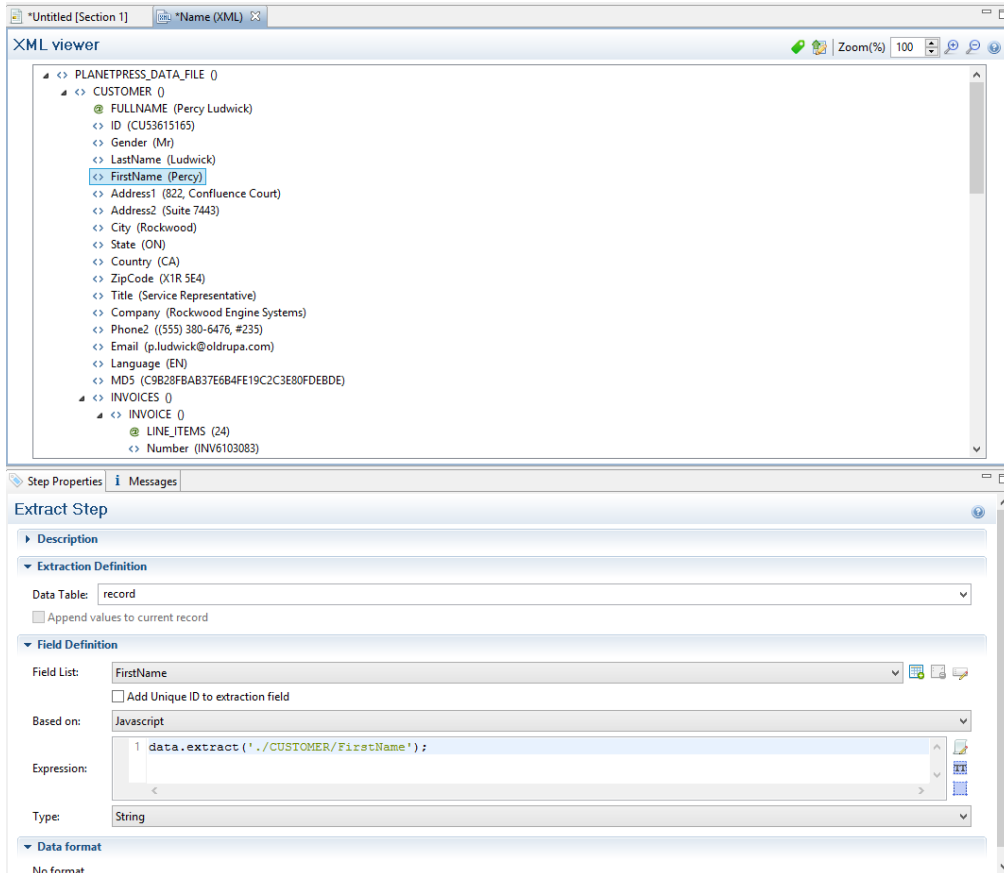
Extracts the text value of the specified node in an **XML** file.

xPath

String that can be relative to the current location or absolute from the start of the record.

Example

The script command **data.extract('./CUSTOMER/FirstName')**; means that the extraction is made on the **FirstName** node under **Customer**.



`extract(columnName, rowOffset)`

Extracts the text value from the specified column and row.

columnName

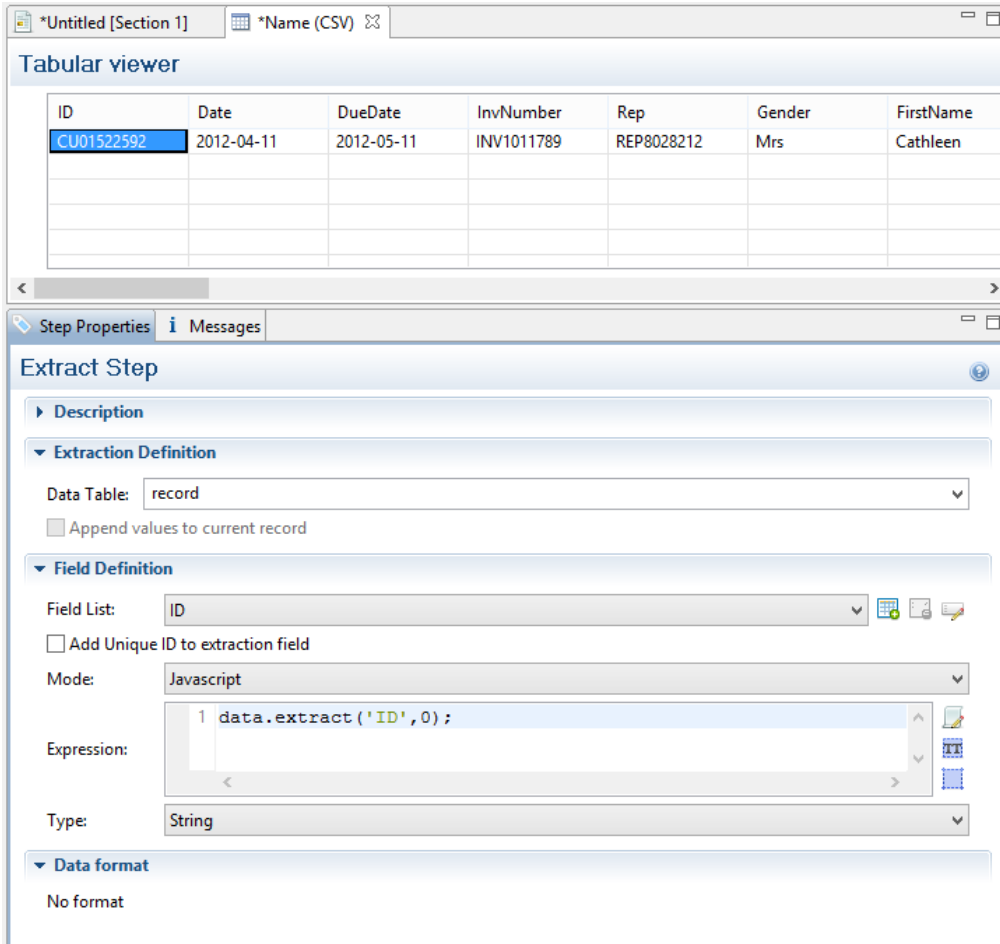
String that represents the column name.

rowOffset

Number that represents the row index (zero-based), relative to the current position. To extract the current row, specify 0 as the rowOffset. Use `moveTo()` to move the pointer in the source data file (see "Example" on page 290).

Example

The script command `data.extract('ID',0)`; means that the extraction is made on the **ID** column in the first row.



`extract(left, right, verticalOffset, lineHeight, separator)`

Extracts the text value from a rectangular region in a **PDF** file. All coordinates are expressed in millimeters.

left

Double that represents the distance from the left edge of the page to the left edge of the rectangular region.

right

Double that represents the distance from the left edge of the page to the right edge of the rectangular region.

verticalOffset

Double that represents the distance from the current vertical position.

lineHeight

Double that represents the total height of the region.

separator

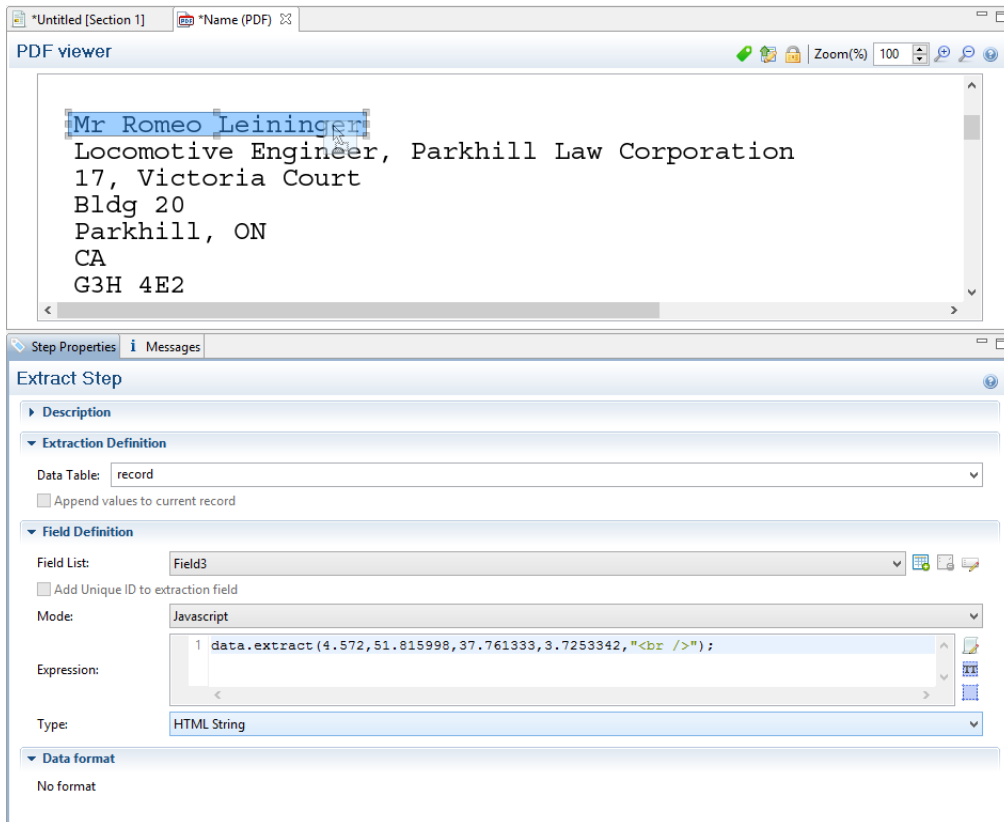
String inserted between all lines returned from the region. If you don't want anything to be inserted between the lines, specify an empty string ("").

Tip

"
" is a very handy string to use as a separator. When the extracted data is inserted in a Designer template, it will be interpreted as a line break, because
 is a line break in HTML and Designer templates are actually HTML files.

Example

The script command **data.extract(4.572,51.815998,37.761333,3.7253342,"
");** means that the left position of the extracted information is located at 4.572mm, the right position at 51.815998mm, the vertical offset is 37.761333mm and the line height is 3.7253342mm. Finally, the "
" string is used for concatenation.



extractMeta()

Method that extracts the value of a metadata field on a certain level in a PDF/VT. This method always return a String.

extractMeta(levelName String, propertyName String)

levelName

String, specifying the PDF/VT's level. Case-sensitive.

propertyName

String, specifying the metadata field.

fieldExists()

Method of the `data` object that returns **true** if a certain metadata field, column or node exists. (See "data" on page 269.)

fieldExists(levelName, propertyName)

This method returns **true** if the given metadata field exists at the given level in a **PDF** file.

levelName

String that specifies the metadata field.

propertyName

String that specifies the level.

fieldExists(fieldName)

This method returns **true** if the specified column exists in the current record in a **CSV** file.

fieldName

String that represents a field name (column) in a CSV file.

fieldExists(xpath)

This method returns **true** if the specified node exists in the current record in an **XML** file.

xPath

String that specifies a node.

find()

Method of the data object that finds the first occurrence of a string starting from the current position.

find(stringToFind, leftConstraint, rightConstraint)

Finds the first occurrence of a string starting from the current position. The search can be constrained to a series of characters (in a text file) or to a vertical strip (in a PDF file) located between the given constraints.

The method returns `null` if the string cannot be found. Otherwise it returns a `RectValueText` (if the data source is a text file) or `RectValuePDF` (if the data source is a PDF file) object. This object contains the absolute Left, Top, Right and Bottom coordinates of the smallest possible rectangle that completely encloses the first occurrence of the string. The coordinates are expressed in a number of characters if the data source is a text file, or in millimetres if the data source is a PDF file.

Partial matches are not allowed. The entire string must be found between the two constraint parameters.

The `data.find()` function only works on the current page. If the record contains several pages, you must create a loop that will perform a jump from one page to another to do a `find()` on each page.

Note

Calling this method does not move the current position to the location where the string was found. This allows you to use the method as a look-ahead function without disrupting the rest of the data mapping workflow.

stringToFind

String to find.

leftConstraint

Number indicating the left limit from which the search is performed. This is expressed in characters for a text file, or in millimetres for a PDF file.

rightConstraint

Number indicating the right limit to which the search is performed. This is expressed in characters for a text file, or in millimetres for a PDF file.

Examples

To look for the word "text" on an entire Letter page (8 1/2 x 11 inch), the syntax is:

```
data.find("text", 0, 216);
```

The numbers 0 and 216 are in millimeters and indicate the left and right limits (constraints) within which the search should be performed. In this example, these values represent the entire width of a page. Note that the smaller the area is, the faster the search is. So if you know that the word "text" is within 3 inches from the left edge of the page, provide the following:

```
data.find("text", 0, 76.2); //76.2mm = 3*25.4 mm
```

The return value of the function is:

Left=26,76, Top=149.77, Right=40,700001, Bottom=154.840302

These values represent the size of the rectangle that encloses the string in full, in millimeters relative to the upper left corner of the current page.

findRegExp()

Finds the first occurrence of a string that matches the given regular expression pattern, starting from the current position.

findRegExp (regexpToFind, flags, leftConstraint, rightConstraint): rectValueText

Finds the first match for a given regular expression pattern starting from the current position. Regular expression flags (**i,s,L,m,u,U,d**) are specified in the flags parameter. The search can be constrained to a vertical column of characters located between the left and right constraint, each expressed in characters (in a text file) or millimeters (in a PDF file). Partial matches are not allowed. The entire match for the regular expression pattern must be found between the two constraints.

The method returns `null` if the regular expression produces no match. Otherwise it returns a `RectValueText` object, containing the Left, Top, Right and Bottom coordinates - expressed in characters (in a text file) or millimeters (in a PDF file), relative to the upper left corner of the current page - of the smallest possible rectangle that completely encloses the first match for the regular expression.

Note

Calling this method does not move the current position to the location where the match occurred. This allows you to use the method as a look-ahead function without disrupting the rest of the data mapping workflow.

regexpToFind

Regular expression pattern to find.

flags

i: Enables case-insensitive matching. By default, case-insensitive matching assumes that only characters in the US-ASCII charset are being matched. Unicode-aware case-insensitive

matching can be enabled by specifying the `UNICODE_CASE` flag (**u**) in conjunction with this flag.

s: Enables `dotall` mode. In `dotall` mode, the expression `.` matches any character, including a line terminator. By default this expression does not match line terminators.

L: Enables literal parsing of the pattern. When this flag is specified, then the input string that specifies the pattern is treated as a sequence of literal characters. Metacharacters or escape sequences in the input sequence will be given no special meaning. The `CASE_INSENSITIVE` (**i**) and `UNICODE_CASE` (**u**) flags retain their impact on matching when used in conjunction with this flag. The other flags become superfluous.

m: Enables multiline mode. In multiline mode, the expressions `^` and `$` match just after or just before, respectively, a line terminator or the end of the input sequence. By default, these expressions only match at the beginning and the end of the entire input sequence.

u: Enables Unicode-aware case folding. When this flag is specified, then case-insensitive matching, when enabled by the `CASE_INSENSITIVE` flag (**i**), is done in a manner consistent with the Unicode Standard. By default, case-insensitive matching assumes that only characters in the US-ASCII charset are being matched.

U: Enables the Unicode version of Predefined character classes and POSIX character classes. When this flag is specified, then the (US-ASCII only) Predefined character classes and POSIX character classes are in conformance with Unicode Technical Standard #18: Unicode Regular Expression Annex C: Compatibility Properties.

d: Enables Unix lines mode. In this mode, only the `\n` line terminator is recognized in the behavior of `.`, `^`, and `$`.

leftConstraint

Number indicating the left limit from which the search is performed. This is expressed in characters for a text file, or in millimeters for a PDF file.

rightConstraint

Number indicating the right limit to which the search is performed. This is expressed in characters for a text file, or in millimeters for a PDF file.

Examples

```
data.findRegExp(/\d{3}-[A-Z]{3}/, "gi", 50, 100);
```

or

```
data.findRegExp("\\d{3}-[A-Z]{3}", "gi", 50, 100);}}
```

Both expressions would match the following strings: 001-ABC, 678-xYz.

Note how in the second version, where the regular expression is specified as a string, some characters have to be escaped with an additional backslash, which is standard in JavaScript.

db

Object that allows to connect to a database.

Methods

The following table describes the methods of the db object.

Method	Description	Available in	File type
connect()	Method that returns a new database connection object.	Boundaries Preprocessor, Extract, Condition, Repeat, Action, and Postprocessor steps	all

connect()

Method that returns a new database connection object.

[connect\(url, user, password\)](#)

This method returns a new database connection object after connecting to the given URL and authenticating the connection with the provided user and password information.

url

String that represents the url to connect to.

user

String that represents the user name for authentication.

password

String that represents the password for authentication.

logger

Global object that allows logging messages such as error, warning or informational messages.

Methods

The following table describes the methods of the logger object.

Method	Parameters	Description
error()	message: string	Logs an error message
info()	message: string	Logs an informational message
warn()	message: string	Logs a warning message

record

The current record in the main data set.

Properties

Property	Return Type
fields	The field values that belong to this record. You can access a specific field value using either a numeric index or the field name,
index	The one-based index of this record, or zero if no data is available.
tables	The details table that belong to this record. You can access a specific table using a numeric index or the table name.

Example

See this How-to for an example of how the current **record index**, and/or the total number of records in the record set, can be displayed in a document: [How to get the record index and count](#).

region

The region object defines a sub-section of the input data. Its properties vary according to the type of data.

This object is available when triggering document boundaries **On script**; see "Setting boundaries using JavaScript" on page 257.

Methods

The following table describes the methods of the region object. This object is available in Boundaries scripts, with all file types.

Method	Description	Return Type
found	Field that contains a boolean value indicating if the last call to <code>boundaries.find()</code> was successful. Since the <code>find()</code> method always returns a region, regardless of search results, it is necessary to examine the value of <code>found</code> to determine the actual result of the operation.	Boolean
range	Read-only object containing the physical coordinates of the region.	Physical location of the region: x1 (left), y1 (top), x2 (right), y2 (bottom), expressed in characters for a text file or in millimeters for a PDF file. For a CSV file, it is the name of the column that defines the region.
createRegion	Creates a <code>region</code> by setting the	A <code>region</code> that has the specified

Method	Description	Return Type
()	physical coordinates of the region object.	coordinates.

createRegion()

This method sets the physical coordinates of the `region` object. The `region` is available when setting document boundaries using a script (see "region" on the previous page).

PDF and Text: createRegion(x1, y1, x2, y2)

Creates a region from the data, using the specified **left** (x1), **top** (y1), **right** (x2) and **bottom** (y2) parameters, expressed in characters for a text file or in millimeters for a PDF file.

x1

Double that represents the left edge of the region.

y1

Double that represents the top edge of the region.

x2

Double that represents the right edge of the region.

y2

Double that represents the bottom edge of the region.

Example

The following script attempts to match `((n,m))` or `((n))` against any of the strings in the specified region and if it does, a document boundary is set.

```
var myRegion = region.createRegion(170,25,210,35);
var regionStrings=boundaries.get(myRegion);
if (regionStrings) {
    for (var i=0;i<regionStrings.length; i++) {
        if (regionStrings[i].match(/\({2}n,*m*\){2}/gi)){
            boundaries.set();
        }
    }
}
```

```
}  
}
```

(The `match()` function expects a regular expression; see w3schools.com.)

CSV or database: createRegion(columnName)

Creates a region from the data in a CSV file, using the specified **columnName** parameter.

columnName

String containing the name of the column where the region is to be created.

Example

This script checks the first value in a certain column. If it is not the same value as in the previous record(s), a document boundary is set.

```
if(!(boundaries.Eof || boundaries.Bof)){  
    var recordValue = boundaries.get(region.createRegion('ID'))[0];  
    if(!(recordValue==boundaries.getVariable('lastValue'))){  
        boundaries.setVariable('lastValue',recordValue);  
        boundaries.set(0);  
    }  
}
```

sourceRecord

Returns a **sourceRecord** object containing properties specific to the current source record being processed.

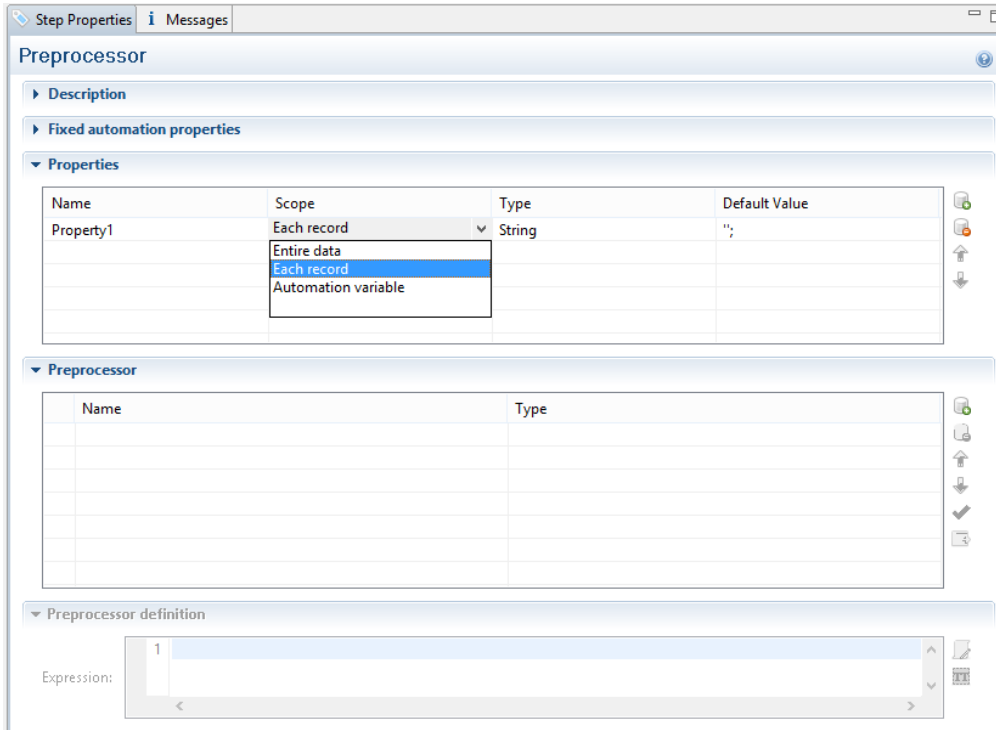
Properties

```
sourceRecord.properties.property;
```

Property	Return Type
properties	Returns an array of properties defined in the Preprocessor step with the Record Scope (i.e. dynamically reset with each new record).

Example

The **property**, used by the object Source Record, must first be declared in a **Preprocessor** step:



1. Enter the property **Name**.
2. Select **Each record** from the Scope drop-down list.
3. Select a **Type** for the Property.

steps

Returns a **steps** object encapsulating properties and methods pertaining to the current DataMapper process.

This object is available in an Extract, Condition, Repeat or Multiple Conditions step script.

Methods and properties

The following table lists the methods and properties of the **steps** object. These are available in Extract, Condition, Repeat, and Action steps, depending on the file type.

Method	Description	File type
currentPosition	Returns the current position of the pointer in the data. Depending on the type of data being processed, the return value may be a string (e.g. XPath value in XML), an integer (e.g. line numbers in text or tabular data), or a measure in millimeters(e.g. PDF data).	All
currentLoopCounter	An integer value representing the current iteration of the containing loop. When loops are nested, you have access to the iteration for the current loop but not to any of the parent loops.	All
currentPage	Returns an integer value representing the current page where the current position is located, inside the current record.	Text, PDF
currentPageHeight	The height of the current page in millimeters.	PDF
currentPageWidth	The width of the current page in millimeters.	PDF
moveTo()	Moves the pointer in the source data file to another position.	All
moveToNext()	Moves the position of the pointer in the source data file to the next line, row or node. The behavior and arguments are different for each emulation type: text, PDF, tabular (CSV), or XML.	All
totalPages	An integer value representing the total number of pages	Text,

Method	Description	File type
	inside the current record.	PDF

Example

```

if(steps.currentPage > curPage) {
    steps.moveTo(0, steps.currentPosition+14); /* Moves the current
position to 14 lines below the current position of the pointer in
the data */
    curPage++;
} else if(curLine.startsWith("LOAD FACTOR")) { /* Extracts data to
the curLine variable until the string "LOAD FACTOR" is encountered
*/
    break;
} else {
    lineArray.push(curLine); /* Adds the current line value
(extraction) to the array */
}

```

moveTo()

Moves the position of the pointer in the source data file. This is a method of the steps object (see "steps" on page 287).

moveTo(scope, verticalPosition)

Moves the current position in a **text file** to `verticalPosition` where the meaning of `verticalPosition` changes according to the value specified for `scope`.

scope

Number that may be set to:

- 0 or `steps.MOVELINES`
- 1 or `steps.MOVEDELIMITERS`
- 2: next line with content

verticalPosition

Number. What it represents depends on the value specified for `scope`.

With the scope set to 0 or `steps.MOVELINES`, `verticalPosition` represents the **index of the line** to move to **from the top of the record**.

With the scope set to 1 or `steps.MOVEDELIMITERS`, `verticalPosition` represents the **index of the delimiter** (as defined in the Input Data settings) to move to **from the top of the record**.

With the scope set to 2, `verticalPosition` is not used. The position is moved to the next line after the current position that contains any text.

Example

The following line of code moves the current position in a text file 14 lines down from the current vertical position (`steps.currentPosition`) of the pointer in the data, as long as it is on the same page.

```
if(steps.currentPage > curPage) {
    steps.moveTo(0, steps.currentPosition+14);
    curPage++;
}
```

`moveTo(scope, verticalOffset)`

Moves the current position in a **PDF file** to `verticalOffset` where the meaning of `verticalOffset` changes according to the value specified for `scope`.

scope

Number that may be set to:

- 0 or `steps.MOVEMEASURE`
- 1 or `steps.MOVEPAGE`

verticalOffset

Double. What it represents depends on the value specified for `scope`.

With the scope set to 0 or `steps.MOVEMEASURE`, `verticalOffset` represents the number of **millimeters** to move the current position, relative to the top of the record (NOT the top of the current page).

With the scope set to 1 or `steps.MOVEPAGES`, `verticalOffset` represents the **index of the target page**, relative to the top of the record.

moveTo(xpath)

Moves the current position in a XML file to the first instance of the given node, relative to the top of the record.

xPath

String that defines a node in the XML file.

Tip

The **XML elements** drop-down (on the Settings pane, under Input Data) lists xPaths defining nodes in the current XML file.

moveTo(row)

Moves the current position in a **CSV file** to the given row number.

row

Number that represents the index of the row, relative to the top of the record.

moveToNext()

Moves the position of the pointer in the source data file to the next line, row or node. The behavior and arguments are different for each emulation type: text, PDF, tabular (CSV), or XML.

This is a method of the steps object (see "steps" on page 287).

moveToNext(scope)

Moves the current position in a **text file** or **XML file** to the next instance of `scope`. What `scope` represents depends on the emulation type: text or XML.

Text

scope

Number that may be set to:

- 0 or `steps.MOVELINES`: the current position is set to the next line.
- 1 or `steps.MOVEDELIMITERS`: the current position is set to the next delimiter (as defined in the Input Data settings).
- 2 (next line with content): the current position is set to the next line that contains any text.

Example

The following line of code moves the current position to the next line that contains any text.

```
steps.moveToNext(2);
```

XML

scope

Number that may be set to:

- 0 or `steps.MOVENODE`: the current position is set to the next parent node in the XML hierarchy.
- 1 or `steps.MOVESIBLING`: the current position is set to the next sibling node in the XML hierarchy.

moveToNext(left, right)

Moves the current position in a **PDF file** to the next line that contains any text, the search for text being contained within the **left** and **right** parameters, expressed in millimeters.

left

Double that represents the left edge (in millimeters) of the text to find.

right

Double that represents the right edge (in millimeters) of the text to find.

moveToNext()

Moves the current position in a **CSV file** to the next row, relative to the current position.

Functions

copyFile()

Function that copies a file to the target file path, replacing it if it already exists.

copyFile(source, target)

source

String that specifies the source file path and name.

target

String that specifies the target file path and name.

Example

This script copies the file test.txt from c:\Content into the c:\out folder.

```
copyFile("c:\Content\test.txt", "c:\out\")
```

createTmpFile()

Function that creates a file with a unique name in the temporary work folder and returns a **file** object. This file stores data temporarily in memory or in a buffer. It is used to prevent multiple input/output access to a physical file when writing. In the end, the contents are transferred to a physical file for which only a single input/output access will occur.

Example

In the following script, the contents of the data sample file are copied in uppercase to a temporary file.

```
try{
    // Open a reader
    var reader = openTextReader(data.filename);
    // Create a temporary file
    var tmpFile = createTmpFile();
    // Open a writer on the temporary file
    var writer = openTextWriter(tmpFile.getPath());
    try{
        var line = null; // Current line
        /* read line by line and readLine will return null at the e
the file */
        while( (line = reader.readLine()) != null ){
            // Edit the line
            line = line.toUpperCase();
            // Write the result in the temporary file
            writer.write(line);
            // add a new line
```

```

        writer.newLine();
    }
}
finally{
    // Close the writer of the temporary file
    writer.close();
}
}
finally{
    // Close the reader
    reader.close();
}
deleteFile(data.filename);
tmpFile.move(data.filename);

```

createHttpRequest()

Function that creates a new `ScriptableHttpRequest` object, in order to issue REST/AJAX calls to external servers.

This feature allows the data mapping process to complement its extraction process with external data, including data that could be provided by an HTTP process in Workflow, for instance a process that retrieves certain values from Workflow's Data Repository. Another possible use is to have a Postprocessor that writes the results of the extraction process to a file and immediately uploads that file to a Workflow process.

The returned `ScriptableHttpRequest` has a selection of the properties and methods of the standard JavaScript `XMLHttpRequest` object (see <https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest>). Supported properties and methods are listed below.

Note

It is not possible to use the **async** mode, which can be set via the `open()` function of the `ScriptableHttpRequest` (see <https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/open>) in a data mapping configuration. Async-related properties and methods of the `ScriptableHttpRequest` object - for example `.onreadystatechange`, `.readyState` and `.onTimeout` - are **not supported**.

The reason for this is that by the time the response comes back from the server, the `DataMapper` script may have finished executing and gone out of scope.

Supported properties

- response
- status
- statusText
- timeout (ms). Default: 1 minute.

Supported methods

create()	Creates a new instance of ScriptableHttpRequest.
<ul style="list-style-type: none">• open(String method, String url, String user, String password)• open(String verb, String url, String userName, String password, String[] headers, String[] headervalues, String requestBody)	Opens a HTTP request. Note If you don't use a user name and password, pass empty strings: <pre>request.open ("GET", url, "", "");</pre>
<ul style="list-style-type: none">• send()• send(String requestBody)	Sends an HTTP request and returns the HTTP status code. Blocked call.
getResponseHeader(String header)	Gets the ResponseHeader by name.
getResponseHeaders()	Returns the full response headers of the last HTTP request.
getRequestBody()	Gets the HTTP request body (for POST and PUT).
setRequestHeader(String requestHeader, String value)	Adds an additional HTTP request header.

<code>getResponseBody()</code>	Returns the full response body of the last HTTP request.
<code>setRequestBody(String requestBody)</code>	Sets the HTTP request body (for POST and PUT).
<code>getPassword()</code>	Gets the password for HTTP basic authentication
<code>setPassword(String password)</code>	Sets the password for HTTP basic authentication
<code>getTimeout()</code>	Gets the time to wait for the server's response
<code>setTimeout(int timeout)</code>	Sets the time (in ms.) to wait for the server's response.
<code>getUsername()</code>	gets the username for basic HTTP authentication.
<code>setUsername(String userName)</code>	sets the username for basic HTTP authentication
<code>abort()</code>	Aborts the request.

deleteFile()

Function that is used to delete a file.

deleteFile(filename)

filename

String that specifies the path and file name of the file to be deleted.

Examples

1. Deleting a file in a local folder:

```
deleteFile("c:\Content\test.txt");
```

2. Deleting the sample data file used in the DataMapper:

```
deleteFile(data.filename);
```

execute()

Function that calls an external program and waits for it to end.

execute(command)

Calls an external program and waits for it to end.

command

String that specifies the path and file name of the program to execute.

newByteArray()

Function that returns a new byte array.

newByteArray(size)

Returns a new byte array of the specified number of elements.

size

Integer that represents the number of elements in the new array.

newCharArray()

Function that returns a new Char array.

newCharArray(size)

Returns a new Char array of the specified number of elements.

size

Integer that represents the number of elements in the new array.

newDoubleArray()

Function that returns a new double array.

newDoubleArray(size)

Returns a new Double array of the specified number of elements.

size

Integer that represents the number of elements in the new array.

newFloatArray()

Function that returns a new float array.

newFloatArray(size)

Returns a new Float array of the specified number of elements.

size

Integer that represents the number of elements in the new array.

newIntArray()

Function that returns a new array of Integers.

newIntArray(size)

Returns a new Integer array of the specified number of elements.

size

Integer that represents the number of elements in the new array.

newLongArray()

Function that returns a new long array.

newLongArray(size)

Returns a new Long array of the specified number of elements.

size

Integer that represents the number of elements in the new array.

newStringArray()

Function that returns a new string array.

newStringArray(size)

Returns a new String array of the specified number of elements.

size

Integer that represents the number of elements in the new array.

openBinaryReader()

Function that opens a file as a binary file for reading purposes. The function returns a BinaryReader object.

openBinaryReader(filename)

filename

String that represents the name of the file to open.

openBinaryWriter()

Function that opens a file as a binary file for writing purposes. The function returns a BinaryWriter object.

openBinaryWriter(filename, append)

filename

String that represents the name of the file to open.

append

Boolean parameter that specifies whether the file pointer should initially be positioned at the end of the existing file (append mode) or at the beginning of the file (overwrite mode).

openTextReader()

Function that opens a file as a text file for reading purposes. The function returns a TextReader object. Please note that the temporary file must be closed at the end.

openTextReader(filename,encoding)

filename

String that represents the name of the file to open.

encoding

String that specifies the encoding of the file to read (UTF-8, ISO-8859-1, etc.).

Example

In the following example, the `openTextReader()` function is used to open the actual data sample file in the Data Mapper for reading.

```
var fileIn = openTextReader(data.filename);
var tmp = createTmpFile();
var fileOut = openTextWriter(tmp.getPath());
var line;

while((line = fileIn.readLine())!=null){
    fileOut.write(line.replace((subject), ""));
    fileOut.newLine();
}

fileIn.close();
fileOut.close();
deleteFile(data.filename);
tmp.move(data.filename);
tmp.close();
```

OpenTextWriter()

This function opens a file as a text file for writing purposes. The function returns a `TextWriter` object. This must be closed at the end.

OpenTextWriter(filename, encoding, append)

filename

String that represents the name of the file to open.

encoding

String specifying the encoding to use (UTF-8, ISO-8859-1, etc.)..

append

Boolean parameter that specifies whether the file pointer should initially be positioned at the end of the existing file (append mode) or at the beginning of the file (overwrite mode).

Example

In the following example, the **openTextWriter** function is used to open the newly created temporary file for writing:

```
var fileIn = openTextReader(data.filename);
var tmp = createTmpFile();
var fileOut = openTextWriter(tmp.getPath());
var line;

while ((line = fileIn.readLine()) != null) {
    fileOut.write(line.replace((subject), ""));
    fileOut.newLine();
}

fileIn.close();
fileOut.close();
deleteFile(data.filename);
tmp.move(data.filename);
tmp.close();
```

The Designer

The Designer is a WYSIWYG (what you see is what you get) editor that lets you create templates for various output channels: Print, Email and Web. A template may contain designs for multiple output channels: a letter intended for print and an e-mail variant of the same message, for example. Content, like the body of the message or letter, can be shared across these contexts. Templates are personalized using scripts and variable data extracted via the DataMapper. More advanced users may use native HTML, CSS and JavaScript.

The following topics will help to quickly familiarize yourself with the Designer.

- "Designer basics" below. These are the basic steps for creating and developing a template.
- "Features" on the next page. These are some of the key features in the Designer.
- "Designer User Interface" on page 666. This part gives an overview of all elements in the Designer User Interface, like menus, dialogs and panes.

More help can be found here:

- [Tutorials](#) On Video: watch an introductory video, overview tutorials or practical how-to videos.
- [Forum](#): Browse the forum and feel free to ask questions about the use of Connect software
- [Demo site](#). Download demonstrations of OL products.

...

Designer basics

With the Designer you can create templates for personalized letters, emails and web pages, and generate output from them.

These are the basic steps for creating and developing a template:

1. **Create a template**

Create a template, using one of the Template Wizards. See "Creating a template" on the facing page.

2. **Fill the template**

Add text, images and other elements to the template and style them. See "Content elements" on page 465 and "Styling and formatting" on page 551.

3. **Personalize the content**

Personalize the content using variable data. See "Personalizing Content" on page 592.

4. **Generate output**

Adjust the settings, test the template and generate output: letters, emails, and/or web pages. See "Generating output" on page 953.

5. **What's next**

Use [Workflow](#) to automate your customer communications.

Note

Steps 2 and 3 are not necessarily to be followed in this order. For example, as you add elements to a template, you may start personalizing them right away, before adding other elements to the template.

Features

The Designer is Connect's module to create templates for personalized customer communications. These are some of the key features in the Designer:

"Templates" on the facing page. Start creating, using and sharing templates.

"Contexts" on page 320. A context contains one or more designs for one output channel:

- "Print" on page 325. This topic helps you design and fill sections in the Print context.
- "Email" on page 359. This topics helps you design an email template.
- "Web" on page 381. This topic helps you design a web page.

"Sections" on page 321. Sections in one context are designed for the same output channel.

"Content elements" on page 465. Elements make up the biggest part of the content of each design.

"Snippets" on page 548. Snippets help share content between contexts, or insert content conditionally.

"Styling and formatting" on page 551. Make your Designer templates look pretty and give them the same look and feel with style sheets.

"Personalizing Content" on page 592. Personalize your customer communications using variable data.

"Writing your own scripts" on page 624. Scripting can take personalization much further. Learn how to script via this topic.

"Generating output" on page 953. Learn the ins and outs of generating output from each of the contexts.

Templates

The Designer is a WYSIWYG (what you see is what you get) tool to create templates. This topic gets you started. It explains how to create a template, what is found in a template file, and how output can be generated.

Creating a template

In the **Welcome** screen that appears after startup, get off to a flying start choosing **Browse Template Wizards**. Scroll down to see all the Template Wizards. After deciding which output channel – print, email or web – will be prevalent in your template, select a template.

The Template Wizards can also be accessed from the menu: click **File**, click **New**, expand the **Template** folder, and then expand one of the **templates** folders.

There are Wizards for the three types of output channels, or **contexts** as they are called in the Designer: Print, Email and Web.

See:

- "Creating an Email template with a Wizard" on page 364
- "Creating a Print template with a Wizard" on page 327

- "Creating a Web template with a Wizard" on page 382

Tip

The quickest way to create a Print template based on a PDF file is to right-click the PDF file in the Windows Explorer and select **Enhance with Connect**.

After creating a template you can add the other contexts (see "Contexts" on page 320), as well as extra sections (see "Sections" on page 321), to the template.

It is, however, not possible to use a Template Wizard when adding a context or section to an existing template.

Tip

If an Email context is going to be part of the template, it is recommended to start with an Email Template Wizard; see "Creating an Email template with a Wizard" on page 364. After creating a template, contexts can be added to it, but that can not be done with a wizard.

Opening a template

To open a template from the Welcome screen, select **Open an Existing Template**.

To open a template from the menu, select **File > Open**.

Then select the template file. A template file has the extension .OL-template.

Warning

A template created in an older version of the software can be opened in a newer version. However, opening and saving it in a newer version of the software will convert the template to the newest file format. The converted template can't be opened in older versions of the software.

Opening a package file

Templates can also be stored in a package file (see "Sharing a template" on page 308). To open a package file, switch the file type to Package files (*.OL-package) in the Open File

dialog. When the package contains print presets, you will be asked if you want to import them into the proper repositories.

Saving a template

A Designer template file has the extension **.OL-template**. It is a zip file that includes up to 3 contexts, all the related resources and scripts, and (optionally) a link to a Data Mapping Configuration.

To save a template for the first time, select **File > Save as**. After that you can save the template by selecting **File > Save** or pressing **Ctrl+S**.

Tip

To quickly copy the name of any other file, set **Save as type** to **Any file (*.*)** in the Save dialog. Select a file to put its name in the File name field. Then set Save as type to Template files (*.OL-template) and save the template.

When more than one resource (template or data mapping configuration) is open and the Designer software is closed, the Save Resources dialog appears. This dialog displays a list of all open resources with their names and file location. Selected resources will be saved, deselected resources will have all their changes since they were last saved dismissed.

Saving older templates

Saving a template in a newer version of the software will convert the template to the newest file format. This makes it unreadable to older versions of the software.

The warning message that is displayed in this case can be disabled.

To re-enable this message (and all other warning dialogs), go to **Window > Preferences > General**, and click the **Reset All Warning Dialogs** button at the bottom.

Associated data mapping configuration

When you save a template, any data mapping configuration that is currently open will be associated with the template by saving a link to the data mapping configuration in the template file.

The next time you open the template you will be asked if you want to open the associated data mapping configuration as well.

To change which data mapping configuration is linked to the template, open both the template and the data mapping configuration that should be linked to it; then save the template.

Auto Save

After a template has been saved for the first time, Connect Designer can auto save the template with a regular interval. To configure Auto Save:

1. Select the menu option **Window > Preferences > Save**.
2. Under **Auto save**, check the option **Enable** to activate the Auto Save function.
3. Change how often it saves the template by typing a number of minutes.

Auto Backup

Connect Designer can automatically create a backup file when you **manually** save a template. To configure Auto Backup:

1. Select the menu option **Window > Preferences > Save**.
2. Under **Auto backup**, check the option **Enable** to activate the Auto Backup function.
3. Type the number of revisions to keep.
4. Select the directory in which the backups should be stored.

Backup files have the same name as the original template with two underscores and a progressive number (without leading zeros) at the end: **originalname__1.OL-template**, **originalname__2.OL-template**, etc.

Note

The Auto Save function does **not** cause backup files to be created.

File properties

On the menu, select **File > Properties** to view and complement the file properties. See File Properties.

The file properties can also be used in scripts; see "template" on page 946. If you are not familiar with writing scripts, refer to "Writing your own scripts" on page 624.

Sharing a template

To share a template, you can send the template file itself, or save the template to a package file, optionally together with a Data Mapping Configuration, a Job Creation Preset and an Output Creation Preset. (See "[Job Creation Presets](#)" on page 840 and "Output Creation Settings" on page 850 for more details.)

To create a package file, select **File > Send to Workflow** and choose **File** in the **Destination** box. For the other options, see "Sending files to Workflow" on the next page. The package file has the extension .OL-package and can be opened in the Designer (see "Opening a package file" on page 305).

Exporting a template report

A template report can be used for archiving purposes or to provide information about the template to people who do not have access to Connect. Such a report can be exported in PDF or XML format. By default it contains a summary of the template with an overview of all the settings and resources that are used in the template: media, master pages, contexts, sections, images, scripts etc. The file properties are included as well (see File Properties).

To open the Export Template Report wizard, select **File > Export Report**. For a description of all options, see Export Template Report wizard.

Creating a custom template report

The Export Template Report wizard also offers the possibility to export custom template reports (in PDF format only). A custom template report could contain another selection of information and present that differently, e.g. with the logo of your company.

To create a custom template report, you need two files:

- A template design with the desired layout and variable data. This .OL-TEMPLATE file has to be made in the Designer.
- A data mapping configuration that provides the variable data. You could use the data mapping configuration made for the standard template report, or create another one in the DataMapper module, using the standard XML template report as data sample.
- Data mapping configurations have the extension .OL-DATAMAPPER.

The following zip file contains both the template and data mapping configuration that are used to generate the standard template report: <http://help.objectiflune.com/en/archive/report-template.zip>.

Generating output from the Designer

Output can be generated directly from the Designer; see "Generating Print output" on page 956, "Generating Email output" on page 973 and "Generating Web output" on page 981.

To test a template first, select **Context > Preflight**. Preflights executes the template without actually producing output and it displays any issues once it's done (see also: "Testing scripts" on page 632).

Sending files to Workflow

Workflow can generate output from a template as well. For this, the template has to be sent to Workflow.

The Send to Workflow dialog sends templates, Data Mapping Configurations and print presets to the Workflow server, or saves them as a package file. Print presets make it possible to do such things as filtering and sorting records, grouping documents and splitting the print jobs into smaller print jobs, as well as the more standard selection of printing options, such as binding, OMR markings and the like. See "**Job Creation Presets**" on page 840 and "Output Creation Settings" on page 850 for more details.

To send one or more templates to Workflow:

1. Select **File > Send to Workflow**.
2. Select the template to send. By default the currently active template is listed. Click **Browse** to select another template. You may select more than one template in the Browse dialog, and each of them is sent to Workflow (or added to a package file). A template file has the extension .OL-template.
3. Select the Data Mapping Configuration to send. By default the current configuration is listed. Click **Browse** to select another configuration. You may select more than one configuration file in the Browse dialog, and each of them is sent to Workflow (or added to a package file). A Data Mapping Configuration file has the extension .OL-datamapper.
4. Use the drop-down to select a Job Creation Preset to send. Click **Browse** to select a preset that is not in the default location for presets. A Job Creation Preset file has the extension .OL-jobpreset.

5. Use the drop-down to select an Output Creation Preset. Click **Browse** to select a preset that is not in the default location for presets. An Output Creation Preset file has the extension .OL-outputpreset.
6. Finally, choose the **Destination**: use the drop-down to select where to send the files. The option **Workflow machines** lists all the PlanetPress Workflow installations detected on the network. Select **File** to save the files as a package that can be loaded within the Workflow tool.

Creating a Web template with a Wizard

With the Designer you can design Web templates and output them through Workflow or as an attachment to an email when generating Email output.

Capture On The Go templates are a special kind of Web templates; see "Capture OnTheGo template wizards" on page 416.

A Web Template Wizard helps you create a Web page that looks good on virtually any browser, device and screen size.

Foundation

All Web Template Wizards in Connect Designer make use of the Zurb **Foundation** front-end framework. A front-end framework is a collection of HTML, CSS, and JavaScript files to build upon. Foundation is a **responsive** framework: it uses CSS media queries and a mobile-first approach, so that websites built upon Foundation look good and function well on multiple devices including desktop and laptop computers, tablets, and mobile phones. Foundation is tested across many browsers and devices, and works back as far as IE9 and Android 2. See <http://foundation.zurb.com/learn/about.html>.

For more information about the use of Foundation in the Designer, see "Using Foundation" on page 420.

After creating a Web template, the other contexts can be added, as well as other sections (see "Adding a context" on page 321 and "Adding a Web page" on page 388).

To create a Web template with a Template Wizard:

1.
 - In the **Welcome** screen that appears after startup, choose **Browse Template Wizards**.
Scroll down until you see the **Foundation Web Page Starter** Template Wizards.
 - Alternatively, on the **File** menu, click **New**, expand the **Template** folder, and then expand the **Foundation Web Page Starter** folder.
2. Select a template. There are 4 types of Web Template Wizards :
 - Blank
 - Contact Us
 - Jumbotron
 - Thank You

If you don't know what template to choose, see "Web Template Wizards" on page 313 further down in this topic, where the characteristics of each kind of template are described.

3. Click **Next** and make adjustments to the initial settings.
 - **Section:**
 - **Name:** Enter the name of the Section in the Web context. This has no effect on output.
 - **Description:** Enter the description of the page. This is the contents of a <meta name="description"> HTML tag.
 - **Top bar** group:
 - **Set width to Grid:** Check this option to limit the width of the top bar contents to the Foundation Grid, instead of using the full width of the page.
 - **Stick to the top of the browser window:** Check to lock the top menu bar to the top of the page, even if the page has scroll bars. This means the menu bar will always be visible in the browser.
 - **Background color:** Enter a valid hexadecimal color code for the page background color (see [w3school's color picker](#)) , or click the colored circle to the right to open the Color Picker.
 - **Colors** group: Enter a valid hexadecimal color code (see [w3school's color picker](#)) or click the colored square to open the Color Picker dialog (see "Color Picker" on page 674), and pick a color for the following elements:

- **Primary:** links on the page.
- **Secondary:** secondary links on the page.
- **Text:** text on the page contained in paragraphs (<p>).
- **Headings:** all headings (<h1> through <h6>) including the heading section's subhead.

4. Click **Finish** to create the template.

The Wizard creates:

- A Web context with one web page template (also called a **section**) in it. The web page contains a Header, a Section and a Footer element with dummy text, and depending on the type of web page, a navigation bar, button and/or Form elements.
- Resources related to the Foundation framework (see "Web Template Wizards" on the next page): style sheets and JavaScript files. The style sheets can be found in the **Stylesheets** folder on the **Resources** pane. The JavaScript files are located in the **JavaScript** folder on the **Resources** pane, in a **Foundation** folder.
- A collection of Snippets in the **Snippets** folder on the Resources pane. The Snippets contain ready-to-use parts to build the web page. Double-click to open them. See "Snippets" on page 548 for information about using Snippets.
- Images: icons, one picture and one thumbnail picture. Hover your mouse over the names of the images in the **Images** folder on the **Resources** pane to get a preview.

The Wizard opens the Web section, so that you can fill it with text and other elements; see "Content elements" on page 465, "Web Context" on page 386 and "Web pages" on page 387.

Web pages can be personalized just like any other type of template; see "Variable Data" on page 604 and "Personalizing Content" on page 592.

Tip

Use the **Outline** pane at the left to see which elements are present in the template and to select an element.

Use the **Attributes** pane at the right to see the current element's ID, class and some other properties.

Use the **Styles** pane next to the Attributes pane to see which styles are applied to the currently selected element.

Tip

Click the **Edges** button on the toolbar to make borders of elements visible on the Design tab. The borders will not be visible on the Preview tab.

Web Template Wizards

Foundation

All Web Template Wizards in Connect Designer make use of the Zurb **Foundation** front-end framework. A front-end framework is a collection of HTML, CSS, and JavaScript files to build upon. Foundation is a **responsive** framework: it uses CSS media queries and a mobile-first approach, so that websites built upon Foundation look good and function well on multiple devices including desktop and laptop computers, tablets, and mobile phones. Foundation is tested across many browsers and devices, and works back as far as IE9 and Android 2. See <http://foundation.zurb.com/learn/about.html>.

Jumbotron

The name of the Jumbotron template is derived from the large screens in sports stadiums. It is most useful for informative or marketing-based websites. Its large banner at the top can display important text and its "call to action" button invites a visitor to click on to more information or an order form.

Contact Us

The Contact Us template is a contact form that can be used on a website to receive user feedback or requests. It's great to use in conjunction with the Thank You template, which can recap the form information and thank the user for feedback.

Thank You

The Thank You template displays a thank you message with some text and media links.

Blank web page

The Blank Web Page template is a very simple Foundation template that contains a top bar menu and some basic contents to get you started.

Capture OnTheGo template wizards

With the Designer you can create Capture OnTheGo (COTG) templates. COTG templates are used to generate forms for the Capture OnTheGo mobile application. For more information about this application, see the website: [Capture OnTheGo](#).

A Capture OnTheGo Form is actually just a Web Form, that you could add without a wizard, but the COTG Template Wizards include the appropriate JavaScript files for the Capture OnTheGo app, and styles to create user-friendly, responsive forms. They are built upon the Foundation framework.

Foundation

All Web Template Wizards in Connect Designer make use of the Zurb **Foundation** front-end framework. A front-end framework is a collection of HTML, CSS, and JavaScript files to build upon. Foundation is a **responsive** framework: it uses CSS media queries and a mobile-first approach, so that websites built upon Foundation look good and function well on multiple devices including desktop and laptop computers, tablets, and mobile phones. Foundation is tested across many browsers and devices, and works back as far as IE9 and Android 2. See <http://foundation.zurb.com/learn/about.html>.

For more information about the use of Foundation in the Designer, see "Using Foundation" on page 420.

After creating a COTG template, the other contexts can be added, as well as other sections (see "Adding a context" on page 321 and "Adding a Web page" on page 388).

Tip

If the COTG Form replaces a paper form, it can be tempting to stick to the original layout. Although that may increase the recognizability, it is better to give priority to the user-friendliness of the form. Keep in mind that the COTG form will be used on a device and don't miss the chance to make it as

user-friendly as possible. See "Designing a COTG Template" on page 413.

Creating a COTG template using a Wizard

To create a COTG template with a Template Wizard:

- In the **Welcome** screen that appears after startup and when you click the Home icon at the top right, choose **Browse Template Wizards**. Scroll down until you see the **Capture OnTheGo Starter** Template Wizards.
 - Alternatively, on the **File** menu, click **New**, expand the **Template** folder, and then expand the **Capture OnTheGo Starter** folder.
2. Select a template. There are 8 types of Web Template Wizards:
 - **Blank**. The Blank COTG Template has some basic design and the appropriate form, but no actual form or COTG elements.
 - **Bill of Lading**. The Bill of Lading Template is a transactional template that includes a detail table with a checkmark on each line, along with Signature and Date COTG elements. Use this wizard as a way to quickly start any new Zurb Foundation based form for Capture OnTheGo.
 - **Event Registration**. The Event Registration Template is a generic registration form asking for name, phone, email, etc.
 - **Event Feedback**. The Event Feedback Template is a questionnaire containing different questions used to rate an experience.
 - **Membership Application**. The Membership Application Template is a signed generic request form that can be used for memberships such as gyms, clubs, etc.
 - **Patient Intake**. The Patient Intake Template is a generic medical questionnaire that could potentially be used as a base for insurance or clinic form.
 - **Kitchen Sink**. The Kitchen Sink Template includes a wide range of basic form and COTG form elements demonstrating various possibilities of the software.
 - **Time Sheet**. The Time Sheet Template is a single page application used to add time entries to a list. This template demonstrates the dynamic addition of lines within a COTG template, as the Add button creates a new time entry. There is no limit to the number of entries in a single page. Submitted data are grouped using arrays (see "Grouping data using arrays" on page 434).

3. Click **Next** and make adjustments to the initial settings.
 - **Create Off-Canvas navigation menu:** an Off-Canvas menu is a Foundation component that lets you navigate between level 4 headings (<h4>) in the form. Check this option to add the menu automatically.
 - **Submit URL:** enter the URL where the form data should be sent. The URL should be a server-side script that can accept COTG Form data.
 - The **Title** and the **Logo** that you choose will be displayed at the top of the Form.
 - **Colors:** Click the colored square to open the Color Picker dialog (see "Color Picker" on page 674) and pick a color, or enter a valid hexadecimal color code (see [w3school's color picker](#)) for the page background color. Do the same for the background color of the navigation bar at the top and for the buttons on the Form.
4. Click **Next** to go to the next settings page if there is one.
5. Click **Finish** to create the template.

The Wizard creates:

- A **Web context** with one web page template (also called a section) in it. The web page contains an 'off-canvas' Div element, Header, a Section and a Footer element with dummy text, and depending on the type of web page, a navigation bar, button and/or Form elements.
- **Style sheets** and **JavaScript files** related to the COTG form itself and others related to the Foundation framework (see above). The style sheets can be found in the Stylesheets folder on the Resources pane. The JavaScript files are located in the JavaScript folder on the Resources pane.
- A collection of **snippets** in the Snippets folder on the Resources pane. The snippets contain ready-to-use parts to build the web form. Double-click to open them. See "Snippets" on page 548 and "Loading a snippet via a script" on page 640 for information about using Snippets.

The Wizard opens the Web section, so that you can fill the Capture OnTheGo form.

6. Make sure to set the action and method of the form: select the form and then enter the action and method on the Attributes pane.

The **action** of a Capture OnTheGo form should specify the Workflow HTTP Server Input task that receives and handles the submitted data. The action will look like this:

http://127.0.0.1:8080/action (8080 is Workflow's default port number; 'action' should be replaced by the HTTP action of that particular HTTP Server Input task).

The **method** of a Capture OnTheGo form should be **POST** to ensure that it doesn't hit a data limit when submitting the form. The GET method adds the data to the URL, and the length of a URL is limited to 2048 characters. Especially forms containing one or more Camera inputs may produce a voluminous data stream that doesn't fit in the URL. GET also leaves data trails in log files, which raises privacy concerns. Therefore POST is the preferred method to use.

Filling a COTG template

Before inserting elements in a COTG Form, have the design ready; see "Designing a COTG Template" on page 413.

In a Capture OnTheGo form, you can use special Capture OnTheGo Form elements, such as a Signature and a Barcode Scanner element. For a description of all COTG elements, see: "COTG Elements" on page 519. To learn how to use them, see "Using COTG Elements" on page 431.

Foundation, the framework added by the COTG template wizards, comes with a series of features that can be very useful in COTG forms; see "Using Foundation" on page 420.

Naturally, Web Form elements can also be used on COTG Forms (see "Forms" on page 527 and "Form Elements" on page 532) as well as text, images and other elements (see "Content elements" on page 465).

Capture OnTheGo templates can be personalized just like any other type of template; see "Variable Data" on page 604 and "Personalizing Content" on page 592.

Tip

Use the **Outline** pane at the left to see which elements are present in the template and to select an element.

Use the **Attributes** pane at the right to see the current element's ID, class and some other properties.

Use the **Styles** pane next to the Attributes pane to see which styles are applied to the currently selected element.

Tip

Click the **Edges** button on the toolbar to make borders of elements visible on the Design tab. The borders will not be visible on the Preview tab.

Resources

This page clarifies the difference between Internal, External and Web resources that may be used in a template, and explains how to refer to them in HTML and in scripts.

Internal resources

Internal resources are files that are added to and saved with the template. To add images, fonts, style sheets, and snippets to your template, you can drag or copy/paste them into the Resources Pane. See also: "Images" on page 537, "Snippets" on page 548, "Styling templates with CSS files" on page 553 and "Fonts" on page 587.

Resource files can also be dragged or copy/pasted **out** of the the application to save them on a local hard drive.

Once imported, internal resources are accessed using a relative path, depending where they're called from. Resources can be located in the following folders:

- images/ contains the files in the Images folder.
- fonts/ contains the files in the Fonts folder.
- css/ contains the files in the StyleSheets folder.
- js/ contains the files in the JavaScripts folder.
- snippets/ contains the files in the Snippets folder.

When referring to them, normally you would simply use the path directly with the file name. The structure within those folders is maintained, so if you create a "signatures" folder within the "Images" folder, you need to use that structure, for example in HTML: ``. In scripts, you can refer to them in the same way, for example:

```
results.loadhtml("snippets/en/navbar.html");
```

See also: "Loading a snippet via a script" on page 640 and "Writing your own scripts" on page 624.

Note

When referring to images or fonts from a CSS file, you need to remember that the current path is `css/`, meaning you can't just call `images/image.jpg`. Use a relative path, for example: `#header { background-image: url('../images/image.jpg'); }`

External resources

External resources are not stored in the template, but on the local hard drive or on a network drive. They are accessed using a path. The path must have forward slashes, for example `` or `var json_variables = loadjson ("file:///d:/jsondata/variables.json");`. The complete syntax is: `file://<host>/<path>`. If the host is "localhost", it can be omitted, as it is in the example, resulting in `file:///<path>`. The empty string is interpreted as 'the machine from which the URL is being interpreted'.

Network paths are similar: `results.loadhtml ("file://servername/sharename/folder/snippet.html");` (note that in this case **file** is followed by 2 slashes only).

Some limitations

- Style sheets cannot refer to external resources.
- The Connect Server user needs access to whichever network path is used. If the network path is on a domain, the Connect Server must be identified with domain credentials that have access to the domain resources.

For more information on network paths, please see this Wikipedia entry: [file URI scheme](#).

Web resources

Web resources are simply accessed using a full URL. This URL needs to be publicly accessible: if you type in that URL in a browser on the server, it needs to be visible. Authentication is possible only through URL Parameters (`http://www.example.com/data.json?user=username&password=password`) or through HTTP Basic Auth (`http://username:password@www.example.com/data.json`).

Resources can also be called from a PlanetPress Workflow instance:

- "Static Resources", as set in the preferences, are accessed using the resource path, by default something like `http://servername:8080/_iRes/images/image.jpg`. (For guidance on setting the preferences, search for 'HTTP Server Input 2' in the PlanetPress Workflow help files on: [OL Help](#)).
- Resources can also be served by processes: `http://servername:8080/my_process?filename=image.jpg` (assuming "my_process" is the action in the HTTP Server Input).

Contexts

Contexts are parts of a template that are each used to generate a specific type of output: Web, Email or Print.

- The Print context outputs documents to either a physical printer a PDF file; see "Print context" on page 332.
- The Email context outputs HTML email, composed of HTML code with embedded CSS. See "Email context" on page 368.
- The Web context outputs an HTML web page. See "Web Context" on page 386.

When a new template is made, the Context appropriate to that new template is automatically created, including one section. After a template has been created, the other two contexts can be added to it; see "Adding a context" on the next page.

Tip

If an Email context is going to be part of the template, it is recommended to start with an Email Template Wizard; see "Creating an Email template with a Wizard" on page 364. After creating a template, contexts can be added to it, but that can not be done with a wizard.

Outputting and combining contexts

All three contexts can be present in any template and they can all be used to output documents; see "Generating Email output" on page 973, "Generating Print output" on page 956 and "Generating Web output" on page 981.

They can even be combined in output.

If present in the same template, a Print context and a Web context can be attached to an Email context.

Outputting other combinations of contexts, and selecting sections based on a value in the data, can be done via a Control Script; see "Control Scripts" on page 645.

Adding a context

To add a context, right-click the **Contexts** folder on the **Resources** pane and click **New print context**, **New email context** or **New web context**. Only one context of each type can be present in a template. Each context, however, can hold more than one section; see "Sections" below.

Deleting a context

To delete a context, right-click the context on the **Resources** pane and click **Delete**.

Warning

No backup files are maintained in the template. The only way to recover a deleted section, is to click **Undo** on the **Edit** menu, until the deleted section is restored. After closing and reopening the template it is no longer possible to restore the deleted context this way.

Sections

Sections are parts of one of the contexts in a template: Print, Email or Web. They contain the main text flow for the contents. In each of the contexts there can be multiple sections. A Print context, for example, may consist of two sections: a covering letter and a policy.

Adding a section

To add a section to a context, right-click the context (Email, Print or Web) on the **Resources** pane, and then click **New section**.

The new section has the same settings as the first section in the same context. However, custom style sheets and JavaScript files aren't automatically included in the new section.

It is not possible to use a Template Wizard when adding a section to an existing template.

Tip

If an Email context is going to be part of the template, it is recommended to start with an Email Template Wizard; see "Creating an Email template with a Wizard" on page 364. After creating a template, contexts can be added to it, but that can not be done with a wizard.

Editing a section

To open a section, expand the **Contexts** folder on the **Resources** pane, expand the respective context (**Print**, **Email** or **Web**) and double-click a section to open it.

Each section can contain text, images and many other elements (see "Content elements" on page 465), including variable data and other dynamic elements (see "Personalizing Content" on page 592).

Copying a section

Copying a section, either within the same template or from another template, can only be done manually. You have to copy the source of the HTML file:

1. Open the section that you want to copy and go to the **Source** tab in the workspace.
2. Copy the contents of the **Source** tab (press **Ctrl+A** to select everything and then **Ctrl+C** to copy the selection).
3. Add a new section (see "Adding a section" on the previous page, above).
4. Go to the **Source** tab and paste the contents of the other section here (press **Ctrl+V**).
5. When copying a section to another template, add the related source files, such as images, to the other template as well.

Deleting a section

To delete a section:

- On the **Resources** pane, expand the **Contexts** folder, expand the folder of the respective context, right-click the name of the section, and then click **Delete**.

Warning

No backup files are maintained in the template. The only way to recover a deleted section, is to click **Undo** on the **Edit** menu, until the deleted section is restored. After closing and reopening the template it is no longer possible to restore the deleted context this way.

Renaming a section

To rename a section:

- On the **Resources** pane, expand the **Contexts** folder, expand the folder of the respective context, right-click the name of the section, and then click **Rename**.

Note

Sections cannot have an integer as name. The name should always include alphanumeric characters.

Section properties

Which properties apply to a section, depends on the context it is part of. See also: "Print sections" on page 335, "Email templates" on page 370, and "Web pages" on page 387.

To change the properties for a section:

- On the **Resources** pane, expand the **Contexts** folder, expand the folder of the respective context, right-click the name of the section, and then click one of the options.

Applying a style sheet to a section

In order for a style sheet to be applied to a specific section, it needs to be included in that section. There are two ways to do this.

Drag & drop a style sheet

1. Click and hold the mouse button on the style sheet on the **Resources** pane.
2. Move the mouse cursor within the **Resources** pane to the section to which the style sheet should be applied.
3. Release the mouse button.

Using the Includes dialog

1. On the Resources pane, right-click the section, then click **Includes**.
2. From the **File types** dropdown, select **Stylesheets**.
3. Choose which CSS files should be applied to this section. The available files are listed at the left. Use the arrow buttons to move the files that should be included to the list at the right.
4. You can also change the order in which the CSS files are read: click one of the included CSS files and use the **Up** and **Down** buttons. Note that moving a style sheet up in the list gives it **less** weight. In case of conflicting rules, style sheets read later will override previous ones.

Note

Style sheets are applied in the order in which they are included in a section. The styles in each following style sheet add up to the styles found in previously read style sheets. When style sheets have a conflicting rule for the same element, class or ID, the **last** style sheet ‘wins’ and overrides the rule found in the previous style sheet.

Arranging sections

Changing the order of the sections in a context can have an effect on how they are outputted; see: "Print sections" on page 335, "Email templates" on page 370 and "Web pages" on page 387.

To rearrange sections in a context:

- On the **Resources** pane, expand the Contexts folder, expand the folder of the respective context, and then drag and drop sections to change the order they are in. Alternatively, right-click a section and click **Arrange**. In the Arrange Sections dialog you can change the order of the sections in the same context by clicking the name of a section and moving it using the **Up** and **Down** buttons.

Outputting sections

Which sections are added to the output, depends on the type of context they are in.

When generating output from the Print context, each of the Print sections is added to the output document, one after the other in sequence, for each record. The sections are added to the output in the order in which they appear on the **Resources** pane. See "Generating Print output" on page 956.

In email and web output, only one section can be executed at a time. The section that will be output is the section that has been set as the 'default'. See "Generating Web output" on page 981 and "Web pages" on page 387 and "Generating Email output" on page 973 and "Email templates" on page 370. The 'default' section is always executed when the template is run using the Create Email Content task in Workflow (see Workflow Help: [Create Email Content](#)).

It is, however, possible to include or exclude sections when the output is generated, or to set another section as the 'default', depending on a value in the data. A Control Script can do this; see "Control Scripts" on page 645.

See "Generating output" on page 953 to learn how to generate Print documents, Web pages or Email.

Print

With the Designer you can create one or more Print templates and merge the template with a data set to generate personal letters, invoices, policies etc.

The Print **context** is the folder in the Designer that can contain one or more Print sections.

Print templates, also called Print *sections*, are part of the Print context. They are meant to be printed to a printer or printer stream, or to a PDF file (see "Generating Print output" on page 956).

The Print context can also be added to Email output as a PDF attachment; see "Generating Email output" on page 973. When generating output from the Print context, each of the Print sections is added to the output document, one after the other in sequence, for each record.

When a Print template is created or when a Print context is added to an existing template the Print context folder is created along with other folders and files that are specific to a Print context (see "Creating a Print template with a Wizard" on the next page, "Adding a context" on page 321 and "Print context" on page 332).

Only one Print section is created at the start, but you can add as many Print sections as you need; see "Print sections" on page 335.

Pages

Unlike emails and web pages, Print sections can contain multiple *pages*. Pages are naturally limited by their size and margins. If the content of a section doesn't fit on one page, the overflow goes to the next page. This happens automatically, based on the section's page size and margins; see "Page settings: size, margins and bleed" on page 344.

Although generally the same content elements can be used in all three contexts (see "Content elements" on page 465), the specific characteristics of pages make it possible to use special elements, such as page numbers; see "Page numbers " on page 345.

See "Pages" on page 343 for an overview of settings and elements that are specific for pages.

Headers, footers, tear-offs and repeated elements (Master page)

In Print sections, there are often elements that need to be repeated across pages, like headers, footers and logos. In addition, some elements should appear on each first page, or only on pages in between the first and the last page, or only on the last page. Examples are a different header on the first page, and a tear-off slip that should show up on the last page.

This is what Master Pages are used for. Master Pages can only be used in the Print context.

See "Master Pages" on page 350 for an explanation of how to fill them and how to apply them to different pages.

Stationery (Media)

When the output of a Print context is meant to be printed on paper that already has graphical and text elements on it (called stationery, or preprinted sheets), you can add a copy of this

media, in the form of a PDF file, to the Media folder.

Media can be applied to pages in a Print section, to make them appear as a background to those pages. This ensures that elements added to the Print context will correspond to their correct location on the preprinted media.

When both Media and a Master Page are used on a certain page, they will both be displayed on the Preview tab of the workspace, the Master Page being 'in front' of the Media and the Print section on top. To open the Preview tab, click it at the bottom of the Workspace or select **View > Preview View** on the menu.

The Media will not be printed, unless this is specifically requested through the printer settings in the Print Wizard; see "Generating Print output" on page 956.

See "Media" on page 353 for further explanation about how to add Media and how to apply them to different pages.

Copy Fit

Copy Fit is a feature to scale text to the available space, the name of a person on a greeting card for example, or the name of a product on a shelf talker. This feature is only available with Box and Div elements in Print sections.

For more information about this feature see "Copy Fit" on page 566.

Creating a Print template with a Wizard

A Print template may consist of various parts, such as a covering letter and a policy. Start with one of the Template Wizards for the first part; other parts can be added later.

To create a Print template with a Template Wizard:

1.
 - In the **Welcome** screen that appears after startup:
 - Choose **Browse Template Wizards** and scroll down until you see the Print Template wizards and select the Postcard or Formal Letter wizard.
 - Or choose **Create a New Template** and select the PDF-based Print wizard.
 - Alternatively, on the **File** menu, click **New**, expand the **Template** folder, and then:

- Select the PDF-based Print wizard.
- Or expand the **Basic Print templates** folder, select Postcard or Formal Letter and click **Next**.

See "Print Template Wizards" below for information about the various types of Template wizards.

2. Make adjustments to the initial settings (the options for each type of template are listed below). Click **Next** to go to the next settings page if there is one.
3. Click **Finish** to create the template.

See "Print context" on page 332 and "Print sections" on page 335 for more information about Print templates.

Tip

Use the **Outline** pane at the left to see which elements are present in the template and to select an element.

Use the **Attributes** pane at the right to see the current element's ID, class and some other properties.

Use the **Styles** pane next to the Attributes pane to see which styles are applied to the currently selected element.

Print Template Wizards

There are three Print Template wizards: one for a formal letter, one for a postcard and one for a Print template based on a PDF that you provide.

Postcard

The Postcard Wizard lets you choose a page size and two background images, one for the front and one for the back of the postcard.

When you click **Finish**, the Wizard creates:

- A Print context with one section in it, that has duplex printing (printing on both sides) enabled. See "Printing on both sides" on page 334.

- Two Master Pages that each contain a background image. The first Master Page is applied to the front of every page in the Print section. The second Master Page is applied to the back of every page in the Print section. See "Master Pages" on page 350.
- **Scripts** and **selectors** for variable data. The **Scripts** pane shows, for example, a script called "first_name". This script replaces the text "@first_name@" on the front of the postcard by the value of a field called "first_name" when you open a data set that has a field with that name. See "Variable Data" on page 604.
- A script called Dynamic Front Image Sample. This script shows how to toggle the image on the front page dynamically. See also "Writing your own scripts" on page 624.
- One empty Media. Media, also called Virtual Stationery, can be applied to all pages in the Print section. See "Media" on page 353.

The Wizard opens the Print section, so that you can fill it with text and other elements; see "Content elements" on page 465. It already has two Positioned Boxes on it: one on the front, for text, and one on the back, for the address.

See "Print context" on page 332 and "Print sections" on page 335 for more information about Print templates.

Formal letter

The Formal Letter Wizard first lets you select the page settings, see "Page settings: size, margins and bleed" on page 344.

These settings are fairly self-explanatory, except perhaps these:

- Duplex means double-sided printing.
- The margins define where your text flow will go. The actual printable space on a page depends on your printer.
- The bleed is the printable space **around** a page. It can be used on some printers to ensure that no unprinted edges occur in the final trimmed document. Printers that can't print a bleed, will misinterpret this setting. Set the bleed to zero to avoid this.
- The number of sections is the number of parts in the Print context. Although this Template wizard can add multiple Print sections to the Print context, it will only add content to the first section.

On the next settings page (click **Next** to go there), you can type a subject, the sender's name and the sender's title. These will appear in the letter. You can also:

- Click the **Browse** button to select a signature image. This image will appear above the sender's name and title.
- Select Virtual Stationery: a PDF file with the letterhead stationery. Also see Media.

When you click **Finish**, the Wizard creates:

- A Print context with one section in it; see "Print context" on page 332 and "Print sections" on page 335.
- One empty Master Page. Master Pages are used for headers and footers, for images and other elements that have to appear on more than one page, and for special elements like tear-offs. See "Master Pages" on page 350.
- One Media. You can see this on the **Resources** pane: expand the **Media** folder. **Media 1** is the Virtual Stationery that you have selected in the Wizard. It is applied to all pages in the Print section, as can be seen in the Sheet Configuration dialog. (To open this dialog, expand the **Contexts** folder on the **Resources** pane; expand the **Print** folder and right-click "Section 1"; then select **Sheet Configuration**.) See "Media" on page 353.
- **Selectors** for variable data, for example: @Recipient@. You will want to replace these by the names of fields in your data. See "Variable Data" on page 604.

The Wizard opens the Print section. You can add text and other elements; see "Content elements" on page 465.

The formal letter template already has an address on it. The address lines are paragraphs, located in one cell in a table with the ID **address-block-table**. As the table has no borders, it is initially invisible. The address lines will stick to the bottom of that cell, even when the address has fewer lines. See "Styling and formatting" on page 551 to learn how to style elements.

Tip

Click the **Edges** button on the toolbar to make borders of elements visible on the Design tab. The borders will not be visible on the Preview tab.

PDF-based Print template

Tip

The quickest way to create a Print template based on a PDF file is to right-click the PDF file in the Windows Explorer and select **Enhance with Connect**.

The PDF-based Print template wizard creates a document from an existing PDF file: a brochure, voucher, letter, etc. The PDF is used as the background image of the Print section (see "Using a PDF file as background image" on page 339). Variable and personalized elements, like a reseller address, voucher codes and so on, can be added in front of it (see "Personalizing Content" on page 592 and "Variable Data" on page 604).

By default, the PDF itself is added to the **Image** folder located in the **Resources** pane. Uncheck the option **Save with template** if the PDF should not be imported in the template. If not saved with the template, the image will remain external. Note that external images need to be available when the template is merged with a record set to generate output, and that their location should be accessible from the machine on which the template's output is produced. External images are updated (retrieved) at the time the output is generated.

After clicking **Next**, you can change the settings for the page. The initial page size and bleed area are taken from the selected PDF.

When you click **Finish**, the Wizard creates:

- A Print context with one section in it; see "Print context" on the facing page and "Print sections" on page 335. The selected PDF is used as the background of the Print section; see "Using a PDF file as background image" on page 339. For each page in the PDF one page is created in the Print section.
- One empty Master Page. Master Pages are used for headers, footers, images and other elements that have to appear on more than one page, and for special elements like tear-offs. See "Master Pages" on page 350.
- One empty Media. Media, also called Virtual Stationery, can be applied to all pages in the Print section. See "Media" on page 353.

Print context

The Print context is the folder in the Designer that can contain one or more Print templates.

Print templates, also called *Print sections*, are part of the Print context. They are meant to be printed to a printer or printer stream, or to a PDF file (see "Generating Print output" on page 956).

The Print context can also be added to Email output as a PDF attachment; see "Generating Email output" on page 973. When generating output from the Print context, each of the Print sections is added to the output document, one after the other in sequence, for each record.

Creating the Print context

You can start creating a Print template with a Wizard (see "Creating a Print template with a Wizard" on page 327), or add the Print context to an existing template (see "Adding a context" on page 321).

Tip

Editing PDF files in the Designer is not possible, but when they're used as a section's background, you can add text and other elements, such as a barcode, to them.

The quickest way to create a Print template based on a PDF file is to right-click the PDF file in the Windows Explorer and select **Enhance with Connect**. Alternatively, start creating a new Print template with a Wizard, using the PDF-based Print template (see "Creating a Print template with a Wizard" on page 327).

To use a PDF file as background image for an existing section, see "Using a PDF file as background image" on page 339.

When a Print template is created, the following happens:

- The Print context is created and one **Print section** is added to it. You can see this on the **Resources** pane: expand the **Contexts** folder, and then expand the **Print** folder. The Print context can contain multiple sections: a covering letter and a policy, for example, or one section that is meant to be attached to an email as a PDF file and another one that is going to be printed out on paper. Only one Print section is added to it at the beginning, but you can add as many print sections as you need; see "Adding a Print section" on page 336. See "Print sections" on page 335 to learn how to fill a Print section.

- One **Master Page** is added to the template, as can be seen on the **Resources** pane, in the **Master Page** folder.

In Print sections, there are often elements that need to be repeated across pages, like headers, footers and logos. In addition, some elements should appear on each first page, or only on pages in between the first and the last page, or only on the last page. Examples are a different header on the first page, and a tear-off slip that should show up on the last page.

This is what Master Pages are used for. Master Pages can only be used in the Print context.

See "Master Pages" on page 350.

Initially, the (empty) master page that has been created with the Print context will be applied to all pages in the Print section, but more Master Pages can be added and applied to different pages.

- One **Media** is added to the template, as is visible on the **Resources** pane, in the **Media** folder. This folder can hold the company's stationery in the form of PDF files. When applied to a page in a Print section, Media can help prevent the contents of a Print section from colliding with the contents of the stationery. See "Media" on page 353 to learn how to add Media and, optionally, print them. Initially, the (empty) media that has been created with the Print context, is applied to all pages in the Print section. You can add more Media and apply them each to different pages.
- One **Stylesheet**, named `context_print_styles.css`, is added to the template, as you can see on the Resources pane, in the **Stylesheets** folder. This stylesheet is meant to be used for styles that are only applied to elements in the Print context. See also "Styling templates with CSS files" on page 553.

Print settings in the Print context and sections

The following settings in the Print context and Print sections have an impact on how the Print context is printed.

Arranging and selecting sections

The Print context can contain one or more Print sections. When generating output from the Print context, each of the Print sections is added to the output document, one after the other in sequence, for each record. The sections are added to the output in the order in which they appear on the **Resources** pane. This order can be changed; see "Print sections" on page 335.

It is also possible to exclude sections from the output, or to include a section only on a certain condition that depends on a value in the data. This can be done using a Control Script; see "Control Scripts" on page 645.

Printing on both sides

To print a Print section on both sides of the paper, that Print section needs to have the Duplex printing option to be enabled; see "Enabling double-sided printing (Duplex, Mixplex)" on page 342. This setting can not be changed in a Job Creation Preset or an Output Creation Preset.

Note

Your printer must support duplex for this option to work.

Setting the binding style for the Print context

The Print context , as well as each of the Print sections, can have its own Finishing settings. In printing, Finishing is the way pages are bound together after they have been printed. Which binding styles can be applied depends on the type of printer that you are using.

To set the binding style of the Print **context**:

1. On the **Resources** pane, expand the **Contexts** folder; then right-click the **Print** context and select **Finishing**.
Alternatively, select **Context > Finishing** on the main menu. This option is only available when editing a Print section in the Workspace.
2. Choose a Binding style and, if applicable, the number of holes. For an explanation of all Binding and Hole making options, see "Finishing Options" on page 841.

To set the binding style of a Print **section**, see "Setting the binding style for a Print section" on page 341.

Overriding binding styles in a job creation preset

A *Job Creation Preset* can override the binding styles set for the Print sections and for the Print context as a whole. To bind output in another way than defined in the template's settings:

1. Create a Job Creation Preset that overrides the settings of one or more sections: select **File > Presets** and see "[Job Creation Presets](#)" on page 840 for more details.
2. Select that Job Creation Preset in the Print wizard; see "Generating Print output" on page 956.

Setting the bleed

The **bleed** is the printable space around a page. It can be used on some printers to ensure that no unprinted edges occur in the final trimmed document. The bleed is one of the settings for a section. See "Page settings: size, margins and bleed" on page 344.

Print sections

Print templates, also called *Print sections*, are part of the Print context. They are meant to be printed to a printer or printer stream, or to a PDF file (see "Generating Print output" on page 956).

The Print context can also be added to Email output as a PDF attachment; see "Generating Email output" on page 973. When generating output from the Print context, each of the Print sections is added to the output document, one after the other in sequence, for each record.

Pages

Unlike emails and web pages, Print sections can contain multiple *pages*. Pages are naturally limited by their size and margins. If the content of a section doesn't fit on one page, the overflow goes to the next page. This happens automatically, based on the section's page size and margins; see "Page settings: size, margins and bleed" on page 344.

Although generally the same content elements can be used in all three contexts (see "Content elements" on page 465), the specific characteristics of pages make it possible to use special elements, such as page numbers; see "Page numbers " on page 345.

See "Pages" on page 343 for an overview of settings and elements that are specific for pages.

Using headers, footers, tear-offs and repeated elements

In Print sections, there are often elements that need to be repeated across pages, like headers, footers and logos. In addition, some elements should appear on each first page, or only on pages in between the first and the last page, or only on the last page. Examples are a different header on the first page, and a tear-off slip that should show up on the last page.

This is what Master Pages are used for. Master Pages can only be used in the Print context.

See "Master Pages" on page 350 for an explanation of how to fill them and how to apply them to different pages.

Using stationery (Media)

When the output of a Print context is meant to be printed on paper that already has graphical and text elements on it (called stationery, or preprinted sheets), you can add a copy of this media, in the form of a PDF file, to the Media folder.

Media can be applied to pages in a Print section, to make them appear as a background to those pages. This ensures that elements added to the Print context will correspond to their correct location on the preprinted media.

Note

When both Media and a Master Page are used on a certain page, they will both be displayed on the Preview tab of the workspace, the Master Page being 'in front' of the Media and the Print section on top. To open the Preview tab, click it at the bottom of the Workspace or select **View > Preview View** on the menu.

See "Media" on page 353 for a further explanation about how to add Media and how to apply them to different pages.

Note: The Media will not be printed, unless this is specifically requested through the printer settings; see "Generating Print output" on page 956.

Copy Fit

Copy Fit is a feature to scale text to the available space, the name of a person on a greeting card for example, or the name of a product on a shelf talker. This feature is only available with Box and Div elements in Print sections.

For more information about this feature see "Copy Fit" on page 566.

Adding a Print section

The Print context can contain multiple sections: a covering letter and a policy, for example, or one section that is meant to be attached to an email as a PDF file and another one that is meant

to be printed out on paper. When a Print template is created (see "Creating a Print template with a Wizard" on page 327 and "Print context" on page 332), only one Print section is added to it, but you can add as many print sections as you need.

To add a section to a context:

- On the **Resources** pane, expand the **Contexts** folder, right-click the **Print** context , and then click **New section**.

The first Master Page (see "Master Pages" on page 350) and Media (see "Media" on page 353) will automatically be applied to all pages in the new section, but this can be changed, see "Applying a Master Page to a page in a Print section" on page 352 and "Applying Media to a page in a Print section" on page 357.

Tip

Editing PDF files in the Designer is not possible, but when they're used as a section's background, you can add text and other elements, such as a barcode, to them.

The quickest way to create a Print template based on a PDF file is to right-click the PDF file in the Windows Explorer and select **Enhance with Connect**. Alternatively, start creating a new Print template with a Wizard, using the PDF-based Print template (see "Creating a Print template with a Wizard" on page 327).

To use a PDF file as background image for an existing section, see "Using a PDF file as background image" on page 339.

Note

Via a Control Script, sections can be added to a Print context dynamically; see "Dynamically adding sections (cloning)" on page 655.

Deleting a Print section

To delete a Print section:

- On the **Resources** pane, expand the **Contexts** folder, expand the **Print** context, right-click the name of the section, and then click **Delete**.

Warning

No backup files are maintained in the template. The only way to recover a deleted section, is to click **Undo** on the **Edit** menu, until the deleted section is restored. After closing and reopening the template it is no longer possible to restore the deleted context this way.

Arranging Print sections

When generating output from the Print context, each of the Print sections is added to the output document, one after the other in sequence, for each record. The sections are added to the output in the order in which they appear on the **Resources** pane, so changing the order of the sections in the Print context changes the order in which they are outputted to the final document.

To rearrange sections in a context:

- On the **Resources** pane, expand the **Print** context and drag and drop sections to change the order they are in.
- Alternatively, on the **Resources** pane, right-click a section in the **Print** context and click **Arrange**. In the Arrange Sections dialog you can change the order of the sections by clicking the name of a section and moving it using the **Up** and **Down** buttons.

Styling and formatting a Print section

The contents of a Print section can be formatted directly, or styled with Cascading Style Sheets (CSS). See "Styling and formatting" on page 551.

In order for a style sheet to be applied to a specific section, it needs to be included in that section. There are two ways to do this.

Drag & drop a style sheet

1. Click and hold the mouse button on the style sheet on the **Resources** pane.
2. Move the mouse cursor within the **Resources** pane to the section to which the style sheet should be applied.
3. Release the mouse button.

Using the Includes dialog

1. On the Resources pane, right-click the section, then click **Includes**.
2. From the **File types** dropdown, select **Stylesheets**.
3. Choose which CSS files should be applied to this section. The available files are listed at the left. Use the arrow buttons to move the files that should be included to the list at the right.
4. You can also change the order in which the CSS files are read: click one of the included CSS files and use the **Up** and **Down** buttons. Note that moving a style sheet up in the list gives it **less** weight. In case of conflicting rules, style sheets read later will override previous ones.

Note

Style sheets are applied in the order in which they are included in a section. The styles in each following style sheet add up to the styles found in previously read style sheets. When style sheets have a conflicting rule for the same element, class or ID, the **last** style sheet ‘wins’ and overrides the rule found in the previous style sheet.

Using a PDF file as background image

In the Print context, a PDF file can be used as a section's background. It is different from the Media in that the section considers the PDF to be content, so the number of pages in the section will be the same as the number of pages taken from the PDF file.

With this feature it is possible to create a Print template from an arbitrary PDF file or from a PDF file provided by the DataMapper. Of course, the PDF file itself can't be edited in a Designer template, but when it is used as a section's background, text and other elements, such as a barcode, can be added to it.

To use a PDF file as background image:

1. On the **Resources** pane, expand the **Print** context, right-click the print section and click **Background**.
2. Click the downward pointing arrow after **Image** and select either **From Datamapper input** or **From PDF resource**.
From Datamapper input uses the active Data Mapping Configuration to retrieve the PDF

file that was used as input file, or another type of input file, converted to a PDF file. With this option you don't need to make any other settings; click OK to close the dialog.

3. For a PDF resource, you have to specify where it is located. Clicking the **Select Image** button opens the Select Image dialog (see "Select Image dialog" on page 725). Click **Resources**, **Disk** or **Url**, depending on where the image is located.
 - **Resources** lists the images that are present in the **Images** folder on the **Resources** pane.
 - **Disk** lets you choose an image file that resides in a folder on a hard drive that is accessible from your computer. Click the **Browse** button to select an image. As an alternative it is possible to enter the path manually. The complete syntax is: file://<host>/<path>. Note: if the host is "localhost", it can be omitted, resulting in file:///<path>, for example: file:///c:/resources/images/image.jpg. Check the option **Save with template** to insert the image into the **Images** folder on the **Resources** pane.
 - **Url** allows you to choose an image from a specific web address. Select the protocol (**http** or **https**), and then enter the web address (for example, <http://www.mysite.com/images/image.jpg>).

Note

It is not possible to use a remotely stored PDF file as a section's background, because the number of pages in a PDF file can not be determined via the http and https protocols. Therefore, with an external image, the option **Save with template** is always checked.

4. Select the PDF's **position**:
 - **Fit to page** stretches the PDF to fit the page size.
 - **Centered** centers the PDF on the page, vertically and horizontally.
 - **Absolute** places the PDF at a specific location on the page. Use the **Top** field to specify the distance between the top side of the page and the top side of the PDF, and the **Left** field to specify the distance between the left side of the page and the left side of the PDF.

5. Optionally, if the PDF has more than one page, you can set the range of **pages** that should be used.

Note

The number of pages in the Print section is automatically adjusted to the number of pages in the PDF file that are being used as the section's background image.

6. Finally, click **OK**.

Note

To set the background of a section in script, you need a Control Script; see "Control Scripts" on page 645 and "Control Script API" on page 930.

Setting the binding style for a Print section

In printing, Finishing is the binding style, or the way pages are bound together. Each Print section can have its own Finishing settings, as well as the Print context as a whole; see "Setting the binding style for the Print context" on page 334.

To set the binding style of a Print **section**:

1. On the **Resources** pane, expand the **Contexts** folder, expand the **Print** context and right-click the Print section.
2. Click **Finishing**.
3. Choose a Binding style and, if applicable, the number of holes.

To set the binding style of the Print **context**, see "Setting the binding style for the Print context" on page 334.

Overriding binding styles in a job creation preset

A *Job Creation Preset* can override the binding styles set for the Print sections and for the Print context as a whole. To bind output in another way than defined in the template's settings:

1. Create a Job Creation Preset that overrides the settings of one or more sections: select **File > Presets** and see "[Job Creation Presets](#)" on page 840 for more details.
2. Select that Job Creation Preset in the Print wizard; see "Generating Print output" on page 956.

Enabling double-sided printing (Duplex, Mixplex)

To print a Print section on both sides of the paper, that Print section needs to have the Duplex printing option to be enabled. This is an option in the Sheet Configuration dialog. (See "Sheet Configuration dialog" on page 726.)

Note

Your printer must support Duplex for this option to work.

To enable Duplex or Mixplex printing:

1. On the **Resources** pane, expand the **Print** context, right-click the print section and click **Sheet configuration**.
2. Check **Duplex** to enable content to be printed on the back of each sheet.
3. When Duplex printing is enabled, further options become available.
 - Check **Omit empty back side for Last or Single sheet** to reset a page to Simplex if it has an empty back side. Thus changing a Duplex job into a **Mixplex** job may reduce volume printing costs as omitted back sides aren't included in the number of printed pages.
Empty means that there is no content and **no master page** on that side. To suppress the master page on empty back sides and single sheets, uncheck the option **Same for all positions** and check the option **Omit Master Page Back in case of an empty back page**.
 - Check **Tumble** to duplex pages as in a calendar.
 - Check **Facing pages** to have the side margins switched alternately, so that after printing and binding the pages, they look like in a magazine or book. See "Pages" on the next page to find out how to set a left and right margin on a page.

Pages

Unlike emails and web pages, Print sections can contain multiple *pages*. Pages are naturally limited by their size and margins. If the content of a section doesn't fit on one page, the overflow goes to the next page. This happens automatically, based on the section's page size and margins; see "Page settings: size, margins and bleed" on the facing page.

Although generally the same content elements can be used in all three contexts (see "Content elements" on page 465), the specific characteristics of pages make it possible to use special elements, such as page numbers; see "Page numbers " on page 345.

The widow/orphan setting lets you control how many lines of a paragraph stick together, when content has to move to another page; see "Preventing widows and orphans" on page 347. You can also avoid or force a page break before or after an entire element, see "Page breaks" on page 349.

Each page in a print section has a natural position: it is the first page, the last page, a 'middle' page (a page between the first and the last page) or a single page. For each of those positions, a different Master Page and Media can be set. A Master Page functions as a page's background, with for example a header and footer. A Media represents preprinted paper that a page can be printed on. See "Master Pages" on page 350 and "Media" on page 353.

Page specific content elements

The specific characteristics of pages make it possible to use these special elements:

- **Page numbers** can only be used in a Print context. See "Page numbers " on page 345 to learn how to add and change them.
- Conditional content and dynamic tables, when used in a Print section, may or may not leave an empty space at the bottom of the last page. To fill that space, if there is any, an image or advert can be used as a **whitespace element**; see "Whitespace elements: using optional space at the end of the last page" on the facing page.
- Dynamic tables can be used in all contexts, but **transport lines** are only useful in a Print context; see "Dynamic table" on page 618.

Positioning and aligning elements

Sometimes, in a Print template, you don't want content to move up or down with the text flow. To prevent that, put that content in a Positioned Box. See "Content elements" on page 465.

When it comes to positioning elements on a page, Guides can be useful, as well as Tables. See "How to position elements" on page 567.

Page settings: size, margins and bleed

On paper, whether it is real or virtual, content is naturally limited by the page size and margins.

These, as well as the bleed, are set per Print section, as follows:

- On the **Resources** pane, right-click a section in the **Print** context and click **Properties**.

For the **page size**, click the drop-down to select a page size from a list of common paper sizes. Changing the width or height automatically sets the page size to Custom.

Margins define where your text flow will go. Static elements can go everywhere on a page, that is to say, within the printable space on a page that depends on the printer.

The **bleed** is the printable space around a page. It can be used on some printers to ensure that no unprinted edges occur in the final trimmed document. Note: Printers that can't print a bleed, will misinterpret this setting. Set the bleed to zero to avoid this.

Tip

By default, measurements settings are in inches (in). You could also type measures in centimeters (add 'cm' to the measurement, for example: 20cm) or in millimeters (for example: 150mm).

To change the default unit for measurement settings to centimeters or millimeters: on the menu, select **Window > Preferences > Print > Measurements**.

Whitespace elements: using optional space at the end of the last page

Print sections with conditional content and dynamic tables (see "Personalizing Content" on page 592) can have a variable amount of space at the bottom of the last page. It is useful to fill the empty space at the bottom with transpromotional material, but of course you don't want extra pages created just for promotional data. 'Whitespace elements' are elements that will only appear on the page if there is enough space for them.

To convert an element into a whitespace element:

1. Import the promotional image or snippet; see "Images" on page 537 and "Snippets" on page 548.
2. Insert the promotional image or snippet in the content.

Note

- Only a top-level element (for example, a paragraph that is not inside a table or div) can function as a whitespace element.
- Do not place the promotional image or snippet inside an absolute positioned box. Whitespace only works for elements that are part of the text flow, not for absolute-positioned boxes.

3. Select the image or the element that holds the promotional content: click it, or use the breadcrumbs, or select it on the **Outline** tab; see "Selecting an element" on page 469.
4. On the **Attributes** pane, check the option **Whitespace element**.
5. (Optional.) Add extra space at the top of the element: on the menu **Format**, click the option relevant to the selected element (Image for an image, Paragraph for a paragraph, etc.) and adjust the spacing (padding and/or margins).
Do not add an empty paragraph to provide space between the whitespace element and the variable content. The extra paragraph would be considered content and could end up on a separate page, together with the whitespace element.

Page numbers

Inserting page numbers

Page numbers can be added to a Print section, but they are usually added to a Master Page, because headers and footers are designed on Master Pages; see also: "Master Pages" on page 350.

To insert a page number, select **Insert > Special character > Markers** on the menu, and then click one of the options to decide with what kind of page number the marker will be replaced:

- **Page number:** The current page number in the document. If a page is empty or does not display a page number, it is still added to the page count.
- **Page count:** The total number of pages in the document, including pages with no contents or without a page number.

- **Content page number:** The current page number in the document, counting only pages with contents that are supplied by the Print section. A page that has a Master Page (as set in the Sheet Configuration dialog, see "Applying a Master Page to a page in a Print section" on page 352) but no contents, is not included in the Content page count.
- **Content page count:** This is the total number of pages in the current document that have contents, supplied by the Print section. A page that has a Master Page but no contents, is not included in the Content page count.
- **Sheet number:** The current sheet number in the document. A sheet is a physical piece of paper, with two sides (or pages). This is equivalent to half the page number, for example if there are 10 pages, there will be 5 sheets.
- **Sheet count:** This marker is replaced by the total number of sheets in the document, whether or not they have contents.

Note

When a marker is inserted, a class is added to the element in which the marker is inserted. Do not delete that class. It enables the software to quickly find and replace the marker when generating output. The respective classes are: `pagenumber`, `pagecount`, `contentpagenumber`, `contentpagecount`, `sheetnumber`, and `sheetcount`.

Tip

Instead of page numbers, you might want to display the current **record index** and/or the total number of records in the record set, in the document. There is a How-to that explains how to do that: [How to get the record index and count](#).

Creating a table of contents

A table of contents can only be created in a template script. The script should make use of the `pageRef()` function. For an example, see "Creating a table of contents" on page 900. If you don't know how to write a script, see "Writing your own scripts" on page 624.

Configuring page numbers

By default the page numbers are Arabic numerals (1, 2, 3, etc.) without leading zeros nor prefix, and page numbering starts with page 1 for each section. But this can be changed. To do that:

1. On the **Resources** pane, right-click a section in the **Print** context and click **Numbering**.
2. Uncheck **Restart Numbering** if you want the page numbers to get consecutive page numbers, instead of restarting the page numbering with this section.

Note

Even if a section is disabled, so it doesn't produce any output, this setting is still taken into account for the other sections. This means that if Restart Numbering is checked on a disabled section, the page numbering will be restarted on the next section.

Disabling a section can only be done in a Control Script (see "Control Scripts" on page 645). Control Scripts can also change where page numbers restart.

3. Use the **Format** drop-down to select uppercase or lowercase letters or Roman numerals instead of Arabic numerals.
4. In **Leading Zeros**, type zeros to indicate how many digits the page numbers should have. Any page number that has fewer digits will be preceded by leading zeros.
5. Type the **Number prefix**. Optionally, check Add Prefix to Page Counts, to add the prefix to the total number of pages, too.
6. Close the dialog.

Preventing widows and orphans

Widows and orphans are lines at the beginning or at the end of a paragraph respectively, dangling at the bottom or at the top of a page, separated from the rest of the paragraph. By default, to prevent orphans and widows, lines are moved to the next page as soon as two lines get separated from the rest of the paragraph. The same applies to list items (in unordered, numbered and description lists).

The number of lines that should be considered a widow or orphan can be changed for the entire Print context, per paragraph and in tables.

Note

Widows and orphans are ignored if the **page-break-inside** property of the paragraph is set to **avoid**; see "Preventing a page break" on page 350.

In the entire Print context

To prevent widows and orphans in the entire Print context:

1. On the menu, select **Edit > Stylesheets**.
2. Select the **Print** context.
3. Click **New** (or, when there are already CSS rules for paragraphs, click the selector **p** and click **Edit**).
4. Click **Format**.
5. After **Widows and Orphans**, type the minimum number of lines that should be kept together.

Alternatively, manually set the **widows** and **orphans** properties in a style sheet:

1. Open the style sheet for the Print context: on the **Resources** pane, expand the **Styles** folder and double-click `context_print_styles.css`.
2. Add a CSS rule, like the following:

```
p { widows: 4; orphans: 3 }
```

Per paragraph

To change the widow or orphan setting for one paragraph only:

1. Open the Formatting dialog. To do this, you can:
 - Select the paragraph using the breadcrumbs or the **Outline** pane (next to the **Resources** pane) and then select **Format > Paragraph** in the menu.
 - Right-click the paragraph and select **Paragraph...** from the contextual menu.
2. After **Widows and Orphans**, type the minimum number of lines that should be kept together.

In tables

The CSS properties **widows** and **orphans** can be used in tables to prevent a number of rows from being separated from the rest of the table.

Detail tables are automatically divided over several pages when needed. A Standard Table doesn't flow over multiple pages by default. Splitting a Standard Table over multiple pages

requires setting the Connect-specific `data-breakable` attribute on all of its rows. You can either open the Source tab, or write a script to replace each `<tr>` with `<tr data-breakable="">`. Note that the effect will only be visible in Preview mode.

To set the number of widows and orphans for a table:

1. Open the Formatting dialog. To do this, you can:
 - Select the table using the breadcrumbs or the **Outline** pane (next to the **Resources** pane) and then select **Format > Table** in the menu.
 - Right-click the paragraph and select **Table...** from the contextual menu.
2. After **Widows** and **Orphans**, type the minimum number of table rows that should be kept together.

Page breaks

A page break occurs automatically when the contents of a section don't fit on one page.

Inserting a page break

To insert a page break before or after a certain element, set the `page-break-before` property or the `page-break-after` property of that element (a paragraph for example; see also "Styling text and paragraphs" on page 562):

1. Select the element (see "Selecting an element" on page 469).
2. On the **Format** menu select the respective element to open the Formatting dialog.
3. In the Breaks group, set the **before** or **after** property.
 - **Before:** Sets whether a page break should occur **before** the element. This is equivalent to the **page-break-before** property in CSS; see [CSS page-break-before property](#) for an explanation of the available options.
 - **After:** Sets whether a page break should occur **after** the element. Equivalent to the `page-break-after` property in CSS; see [CSS page-break-after property](#) for an explanation of the available options.

Click the button **Advanced** to add CSS properties and values to the inline style tag directly. Alternatively you could set this property on the Source tab in the HTML (for example: `<h1 style="page-break-before: always;">`), or add a rule to the style sheet; see "Styling your templates with CSS files" on page 556.

Note

You cannot use these properties on an empty `<div>` or on absolute-positioned elements.

Preventing a page break

To prevent a page break inside a certain element, set the **page-break-inside** property of that element to **avoid**:

- Select the element (see "Selecting an element" on page 469).
- On the **Format** menu, select the respective element to open the Formatting dialog.
- In the **Breaks** group, set the **inside** property to **avoid**, to prevent a page break inside the element. For an explanation of all available options of the **page-break-inside** property in CSS, see [CSS page-break-inside property](#).

Alternatively you could set this property on the Source tab in the HTML (for example: `<ul style="page-break-inside: avoid;">`), or add a rule to the style sheet; see "Styling your templates with CSS files" on page 556.

Adding blank pages to a section

How to add a blank page to a section is described in a how-to: [Create blank page on field value](#).

Master Pages

In Print sections, there are often elements that need to be repeated across pages, like headers, footers and logos. In addition, some elements should appear only on specific pages, such as only the first page, or the last page, or only on pages in-between. Examples are a different header on the first page, and a tear-off slip that shows up on the last page.

This is what Master Pages are used for. Master Pages can only be used in the Print context (see "Print context" on page 332).

Master Pages resemble Print sections, and they are edited in much the same way (see "Editing a Master Page" on the next page) but they contain a single page and do not have any text flow. Only one Master Page can be applied per page in printed output. Then a Print template is created, one master page is added to it automatically. You can add more Master Pages; see

"Adding a Master Page" below. Initially, the original Master Page will be applied to all pages, but different Master Pages can be applied to different pages; see "Applying a Master Page to a page in a Print section" on the facing page.

Examples

There are a few How-tos that demonstrate the use of Master Pages:

- [Showing a Terms and Conditions on the back of the first page only.](#)
- [A tear-off section on the first page of an invoice.](#)
- [Tips and tricks for Media and Master Pages.](#)

Adding a Master Page

When a Print template is created, one master page is added to it automatically. Adding more Master Pages can be done as follows:

- On the **Resources** pane, right-click the **Master pages** folder and click **New Master Page**.
- Type a name for the master page.
- Optionally, set the margin for the header and footer. See "Adding a header and footer" on the facing page.
- Click **OK**.

Initially, the master page that has been created together with the Print context will be applied to all pages in the Print section. After adding more Master Pages, different Master Pages can be applied to different pages; see "Applying a Master Page to a page in a Print section" on the facing page.

Editing a Master Page

Master Pages are edited just like sections, in the workspace. To open a Master Page, expand the **Master pages** folder on the **Resources** pane, and double-click the Master Page to open it.

A Master Page can contain text, images and other elements (see "Content elements" on page 465), including variable data and dynamic images (see "Personalizing Content" on page 592). All elements on a Master Page should have an absolute position or be inside an element that has an absolute position. It is good practice to position elements on a Master Page by placing them in a Positioned Box (see "Content elements" on page 465).

Keep in mind that a Master Page always remains a single page. Its content cannot overflow to a next page. Content that doesn't fit, will not be displayed.

Note

Editing the Master Page is optional. One Master Page must always exist in a Print template, but if you don't need it, you can leave it empty.

Adding a header and footer

Headers and footers are not designed as part of the contents of a Print section, but as part of a Master Page, which is then applied to a page in a print section.

To create a header and footer:

1. First insert elements that form the header or footer, such as the company logo and address, on the Master Page; see "Editing a Master Page" on the previous page.
2. Next, define the margins for the header and footer. The margins for a header and footer are set in the Master Page properties. This does not change the content placement within the Master Page itself; in Master Pages, elements can go everywhere on the page. Instead, the header and footer of the Master Page limit the text flow on pages in the Print sections to which this Master Page is applied. Pages in a Print section that use this Master Page cannot display content in the space that is reserved by the Master Page for the header and footer, so that content in the Print section does not collide with the content of the header and footer. To set a margin for the header and/or footer:
 1. On the **Resources** pane, expand the **Master pages** folder, right-click the master page, and click **Properties**.
 2. Fill out the height of the header and/or the footer. The contents of a print section will not appear in the space reserved for the header and/or footer on the corresponding master page.
3. Finally, apply the master page to a specific page in a print section. See "Applying a Master Page to a page in a Print section" below.

Applying a Master Page to a page in a Print section

Every page in a print section has a natural position: it can be the first page, the last page, one of the pages in between (a 'middle page'), or a single page. For each of those positions, you can

set a different Master Page and Media (see "Media" below). It can even have two master pages, if printing is done on both sides (called duplex printing).

To apply Master Pages to specific page positions in a Print section:

1. On the **Resources** pane, expand the **Print** context; right-click the Print section, and click **Sheet configuration**.
2. Optionally, check **Duplex** to enable content to be printed on the back of each sheet. Your printer must support duplex for this option to work. If Duplex is enabled, you can also check **Tumble** to duplex pages as in a calendar, and **Facing pages** to have the margins of the section switch alternately, so that pages are printed as if in a magazine or book.
3. If the option **Same for all positions** is checked, the same Master Page will be applied to every page in the print section (and to both the front and the back side of the page if duplex printing is enabled). Uncheck this option.
4. Decide which Master Page should be linked to which sheet (position): click the downward pointing arrow after **Master Page Front** and select a Master Page. If Duplex is enabled, you can also select a Master Page for the back of the sheet and consequently, check **Omit Master Page Back in case of an empty back page** to omit the specified Master Page on the last backside of a section if that page is empty and to skip that page from the page count.
5. Optionally, decide which Media should be linked to each sheet.
6. Click OK to save the settings and close the dialog.

Deleting a Master Page

To delete a Master Page, expand the **Master pages** folder on the **Resources** pane, right-click the master page, and click **Delete**.

Note that one Master Page as well as one Media must always exist in a Print template. Just leave it empty if you don't need it.

Media

When the output of a Print context is meant to be printed on paper that already has graphical and text elements on it (called stationery, or preprinted sheets), you can add a copy of this media, in the form of a PDF file, to the Media folder.

Media can be applied to pages in a Print section, to make them appear as a background to those pages. This ensures that elements added to the Print context will correspond to their correct location on the preprinted media.

For further explanation about how to apply Media to different pages, see "Applying Media to a page in a Print section" on page 357.

Media will not be printed, unless you want them to; see below.

Per Media, a front and back can be specified and you can specify on what kind of paper the output is meant to be printed on. This includes paper weight, quality, coating and finishing; see "Setting Media properties" below.

Adding Media

To add a Media, right-click the **Media** folder on the **Resources** pane and select **New Media**.

The new Media is of course empty. You can specify two PDF files for the Media: one for the front, and, optionally, another for the back.

Specifying and positioning Media

Specifying a PDF for the front: the fast way

To quickly select a PDF file for the front of a Media, drag the PDF file from the Windows Explorer to one of the Media. The Select Image dialog opens; select an image and check the option **Save with template** if you want to insert the image into the **Images** folder on the **Resources** pane. (For PDF files selected by URL this option is always checked.)

Alternatively you could first import the PDF file to the **Images** folder on the **Resources** pane (using drag & drop) and drag it from there on one of the Media in the **Media** folder.

Either way, you cannot set any options.

To be able to specify a PDF file for both the front and the back of the Media, and to specify a position for the Media's PDF files, you have to edit the properties of the Media.

Setting Media properties

Media have a number of properties that you can set. You can change the Media's page size and margins (as long as it isn't applied to a section), you can specify a PDF file (or any other

type of image file) for both the front and the back of the Media, and you can determine how the virtual stationery should be positioned on the page. This is done as follows:

1. On the **Resources** pane, expand the **Contexts** folder, expand the **Media** folder, right-click the Media and click **Properties**.
2. Now you can change the name and page size of the Media. Note that it isn't possible to change the page size once the Media is applied to a section. Media can only be applied to sections that have the same size.
3. On the **Virtual Stationery** tab, you can click the **Select Image** button to select a PDF image file.
4. Click **Resources**, **Disk** or **Url**, depending on where the image is located.
 - **Resources** lists the PDF files that are present in the **Images** folder on the **Resources** pane.
 - **Disk** lets you choose an image file that resides in a folder on a hard drive that is accessible from your computer. Click the **Browse** button to select an image. As an alternative it is possible to enter the path manually. The complete syntax is: file://<host>/<path>. Note: if the host is "localhost", it can be omitted, resulting in file:///<path>, for example: file:///c:/resources/images/image.jpg. Check the option **Save with template** to insert the image into the **Images** folder on the **Resources** pane.
 - **Url** allows you to choose an image from a specific web address. Select the protocol (**http** or **https**), and then enter the web address (for example, <http://www.mysite.com/images/image.jpg>).

Note

It is not possible to use a remotely stored PDF file as virtual stationery, because the number of pages in a PDF file can not be determined via the http and https protocols. Therefore, with an external image, the option **Save with template** is always checked.

5. Select a PDF file.
6. If the PDF file consists of more than one page, select the desired page.
7. Click **Finish**.

8. For each of the PDF files, select a **position**:
 - **Fit to page** stretches the PDF to fit the page size.
 - **Centered** centers the PDF on the page, vertically and horizontally.
 - **Absolute** places the PDF at a specific location on the page. Use the **Top** field to specify the distance between the top side of the page and the top side of the PDF, and the **Left** field to specify the distance between the left side of the page and the left side of the PDF.
9. Finally, click **OK**.

Setting the paper's characteristics

To set a Media's paper characteristics:

1. On the **Resources** pane, expand the **Contexts** folder, expand the **Media** folder, and right-click the Media. Click **Characteristics**.
2. Specify the paper's characteristics:
 - **Media Type**: The type of paper, such as Plain, Continuous, Envelope, Labels, Stationery, etc.
 - **Weight**: The intended weight of the media in grammage (g/m²).
 - **Front Coating**: The pre-process coating applied to the front surface of the media, such as Glossy, High Gloss, Matte, Satin, etc.
 - **Back Coating**: The pre-process coating applied to the back surface of the media.
 - **Texture**: The intended texture of the media, such as Antique, Calenared, Linen, Stipple or Vellum.
 - **Grade**: The intended grade of the media, such as Gloss-coated paper, Uncoated white paper, etc.
 - **Hole Name**: A predefined hole pattern that specifies the pre-punched holes in the media, such as R2-generic, R2m-MIB, R4i-US, etc.
3. Click **OK**.

Rename Media

To rename Media:

- On the **Resources** pane, expand the **Contexts** folder, expand the **Media** folder, right-click the Media and click **Rename**. Type the new name and click **OK**.
- Alternatively, on the **Resources** pane, expand the **Contexts** folder, expand the **Media** folder, right-click the Media and click **Properties**. Type the new name in the **Name** field and click **OK**.

Applying Media to a page in a Print section

Every page in a print section has a natural position: it can be the first page, the last page, one of the pages in between (a 'middle page'), or a single page. For each of those positions, you can set different Media.

To apply Media to specific page positions in a Print section:

1. On the **Resources** pane, expand the **Print** context; right-click the Print section, and click **Sheet configuration**.
2. Optionally, check **Duplex** to enable content to be printed on the back of each sheet. Your printer must support duplex for this option to work. If Duplex is enabled, you can also check **Tumble** to duplex pages as in a calendar, and **Facing pages** to have the margins of the section switch alternately, so that pages are printed as if in a magazine or book.
3. If the option **Same for all positions** is checked, the same Media will be applied to every page in the print section. Uncheck this option.
4. Decide which Media should be linked to each sheet position: click the downward pointing arrow after **Media** and select a Media.
5. Optionally, decide which Master Page should be linked to each sheet; see "Master Pages" on page 350.

Note

When both Media and a Master Page are used on a certain page, they will both be displayed on the Preview tab of the workspace, the Master Page being 'in front' of the Media and the Print section on top. To open the Preview tab, click it at the bottom of the Workspace or select **View > Preview View** on the menu.

Dynamically switching the Media

In addition to applying Media to sheets via the settings, it is possible to change Media dynamically, based on a value in a data field, in a script. The script has already been made; you only have to change the name of the Media and the section in the script, and write the condition on which the Media has to be replaced.

1. On the **Resources** pane, expand the **Contexts** folder, expand the **Print** context, right-click the print section and click **Sheet configuration**.
2. Decide which pages should have dynamically switching media: every first page in the Print section, every last page, one of the pages in between (a 'middle page'), or a single page. (Uncheck the option **Same for all positions**, to see all page positions.)
3. In the area for the respective sheet position, click the **Edit script** button next to **Media**.

The Script Wizard appears with a standard script:

```
results.attr("content", "Media 1");
```

Media 1 will have been replaced with the name of the media selected for the chosen sheet position.

The field **Selector** in the Script Wizard contains the name of the section and the sheet position that you have chosen.

4. Change the script so that on a certain condition, another media will be selected for the content. For instance:

```
if(record.fields.GENDER === 'M') {  
    results.attr("content", "Media 2");  
}
```

This script changes the media to Media 2 for male customers.

See "Writing your own scripts" on page 624 if you are not familiar with how scripts are written.

5. Click **Apply**, open the tab **Preview** and browse through the records to see if the script functions as expected.
6. When you click **OK**, the script will be added to the **Scripts** pane.

Rotating the Media in a Print section

The actual orientation of the Media and that of a section to which the Media is applied may not match.

The Media can therefore be rotated per Print section:

- On the **Resources** pane, expand the **Print** context; right-click the Print section, and click **Sheet configuration**.
- Click one of the options next to **Media rotation**.

The Media (to be more accurate: the Virtual Stationery images specified for this Media) as well as the section's background image will be rotated accordingly in the entire section.

Note that any Virtual Stationery settings made for the Media also influence how the Media is displayed in each section (see "Setting Media properties" on page 354).

If in the Media properties, the Virtual Stationery position is set to Absolute, any offset given by the Top and Left values will be applied **after** rotation. A Virtual Stationery image located absolutely at the top left (Top: 0, Left: 0) will still appear at the top left of the page after rotating the Media.

Printing virtual stationery

Media are not printed, unless you want them to. Printing the virtual stationery is one of the settings in a Job Creation Preset. To have the virtual stationery printed as part of the Print output:

1. Create a job creation preset that indicates that Media has to be printed: select **File > Presets** and see "[Job Creation Presets](#)" on page 840 for more details.
2. Select that job creation preset in the Print Wizard; see "Generating Print output" on page 956.

Email

With the Designer you can create one or more Email templates and merge the template with a data set to generate personalized emails.

The Email **context** is the folder in the Designer that can contain one or more Email templates, also called Email **sections**. The HTML generated by this context is meant to be compatible with as many clients and as many devices as possible.

Email template

It is strongly recommended to start creating an Email template with a Wizard; see "Creating an Email template with a Wizard" on page 364. Designing HTML email that displays properly on a variety of devices and screen sizes is challenging. Building an email is not like building for the web. While web browsers comply with standards (to a significant extent), email clients do not. Different email clients interpret the same HTML and CSS styles in totally different ways.

When an Email template is created, either with a Wizard or by adding an Email context to an existing template (see "Adding a context" on page 321), the Email context folder is created along with other files that are specific to an Email context; see "Email context" on page 368.

Only one Email section is created at the start, but you can add as many Email sections as you need; see "Email templates" on page 370. However, when the Designer merges a data set to generate output from the Email context, it can merge only one of the templates with each record; see "Generating Email output" on page 973.

Email templates are personalized just like any other template; see "Variable Data" on page 604.

Sending email

When the template is ready, you can change the email settings (see "Email header settings" on page 373) and send the email directly from the Designer or via Workflow. To test a template, you can send a test email first.

Output, generated from an Email template, can have the following attachments:

- The contents of the Print context, in the form of a single PDF attachment.
- The output of the Web context, as an integral HTML file.
- Other files, an image or a PDF leaflet for example.

Attaching the Print context and/or the Web context is one of the options in the Send (Test) Email dialog.

See "Email attachments" on page 379 and "Generating Email output" on page 973.

Designing an Email template

With the Designer you can design Email templates. It is strongly recommended to start creating an Email template with an Email Template Wizard, because it is challenging to design HTML email that looks good on all email clients, devices and screen sizes that customers use when they are reading their email.

This topic explains why designing HTML email design is as challenging as it is, which solutions are used in the Email Template Wizards and it lists good practices, for example regarding the use of images in HTML email. It will help you to create the best possible Email templates in the Designer.

HTML email challenges

Creating HTML email isn't like designing for the Web. That's because email clients aren't like web browsers. Email clients pass HTML email through a preprocessor to remove anything that could be dangerous, introduce privacy concerns or cause the email client to behave unexpectedly. This includes removing javascript, object and embed tags, and unrecognized tags. Most preprocessors are overly restrictive and remove anything with the slightest potential to affect the layout of their email client. Next, the HTML has to be rendered so that it is safe to show within the email client. Unfortunately, desktop, webmail, and mobile clients all use different rendering engines, which support different subsets of HTML and CSS. More often than not, the result of these operations is that they completely break the HTML email's layout.

Designing HTML email in PlanetPressDesigner

The problem of HTML email is that preprocessing and rendering engines break the HTML email's layout. HTML tables, however, are mostly left untroubled. As they are supported by every major email client, they are pretty much the only way to design HTML emails that are universally supported. That's why Tables are heavily used to position text and images in HTML email.

Nesting tables (putting tables in table cells) and applying CSS styles to each table cell to make the email look good on all screen sizes is a precision work that can be a tedious and demanding. Connect's Designer offers the following tools to make designing HTML email easier.

Email templates: Slate and others

The most obvious solution offered in the Designer is to use one of the templates provided with the Designer; see "Creating an Email template with a Wizard" on page 364. The layout of these templates has been tested and proven to look good in any email client, on any device and screen size. The Tables in these templates are nested (put inside another table) and they have no visible borders, so readers won't notice them.

Tip

Click the **Edges** button on the toolbar to make borders of elements visible on the Design tab. The borders will not be visible on the Preview tab or in the output.

Emmet

Emmet is a plugin that enables the lightning-fast creation of HTML code through the use of a simple and effective shortcut language. The Emmet functionality is available in the HTML and CSS source editors of Connect Designer. Emmet transforms abbreviations for HTML elements and CSS properties to the respective source code. The expansion of abbreviations is invoked with the **Tab** key.

In the Source tab of the Workspace, you could for example type `div.row`. This is the abbreviation for a `<div>` element with the class `row`. On pressing the Tab key, this abbreviation is transformed to:

```
<div class="row"></div>
```

To quickly enter a table with the ID 'green', one row, and two cells in that row, type:

```
table#green>tr>td*2
```

On pressing the Tab key, this is transformed to:

```
<table id="green">
  <tr>
    <td></td>
    <td></td>
  </tr>
</table>
```

All standard abbreviations can be found in Emmet's documentation: [Abbreviations](#).

To learn more about Emmet, please see their website: [Emmet.io](http://emmet.io) and the Emmet.io documentation: <http://docs.emmet.io/>.

Preferences

To change the way Emmet works in the Designer, select **Window > Preferences**, and in the Preferences dialog, select **Emmet**; see "Emmet Preferences" on page 706.

Using CSS files with HTML email

Email clients do not read CSS files and some even remove a <style> tag when it is present in the email's header. Nevertheless, CSS files can be used with the Email context in the Designer. When generating output from the Email context, the Designer converts all CSS rules that apply to the content of the email to inline style tags, as if local formatting was applied.

Using images in email campaigns: tips

Host images on a public server

In the Designer you can add images as resource to the template document. When used in email messages these images are automatically embedded on sending the email. These embedded images appear instantly when viewing the message in your email client. There is, however, a downside to this method: embedded images can't be used to track email open rates. Email services like mandrillapp.com embed a tiny tracer image at the bottom of your message. Each time a recipient opens the email the tracer image (aka beacon image) is downloaded and yet another 'open' is registered. On mobile devices this happens when the user clicks the Display Images button. So, when tracking open rates in your email campaigns, store your images on a publicly-accessible server (preferably your own server - you could set up a process in Workflow to serve images and track open rates) or a reputable image hosting service, like photobucket.com. Don't forget to set the Alternate Text for your images on the Attributes pane.

Do not capture your email in one big image

Most e-mail clients do not automatically download images, so do not capture your email in one big image. The recipient initially sees a blank message and probably deletes it right away.

Do not resize images in your email

Many mail clients do not support image resizing and will show the image in its original dimensions. Resize the images before you link to or embed them.

Use background images wisely

Most mail clients do not support background images: a very good reason to stay away from them in your mainstream email campaign. There is one situation in which they do come in handy. Both iPhone and Android default mail have solid CSS support and cover most of the mobile marketplace. You could use background images to substitute images when viewed on these devices. This is done by hiding the actual image and showing a mobile-friendly image as background image instead. This is a technique used in Responsive Email Design.

Creating an Email template with a Wizard

With the Designer you can design Email templates as well as PDF attachments. PDF attachments are designed in the Print context; see "Print context" on page 332.

It is strongly recommended to start creating an Email template with a Wizard, because designing HTML email that displays properly on a variety of devices and screen sizes is challenging. Email clients can, and will, interpret the same HTML and (inline) CSS in totally different ways (see "Designing an Email template" on page 361).

With an Email Template Wizard you can easily create an Email template that outputs emails that look good on virtually any email client, device and screen size.

After creating an Email template, the other contexts can be added to it, as well as other sections (see "Contexts" on page 320 and "Email templates" on page 370).

To create an Email template with a Template Wizard:

1. In the **Welcome** screen that appears after startup:
 - Choose **Browse Template Wizards**.
Scroll down until you see the Email Template Wizards. There are three types of Email Template Wizards:
 - Basic Email templates
 - Banded Email templates
 - Slate: Responsive Email templates by Litmus.
 - Or choose **Create a New Template** and select the Email template. This starts the Basic Action Email wizard.

Alternatively, on the **File** menu, click **New**, and:

- Select Email Template. This starts the Basic Action Email wizard.
- Or expand the **Template** folder, and then expand the **Basic Email templates** folder, the **Banded Email templates** folder, or the **Slate: Responsive Email Templates by Litmus** folder.

See "Email Template Wizards" on the facing page for information about the various types of Template Wizards.

2. Select a template and click **Next**. If you don't know what template to choose, see below; the characteristics of each kind of template are described further down in this topic.
3. Make adjustments to the initial settings (the options for each type of template are listed below). Click **Next** to go to the next settings page if there is one.
4. Click **Finish** to create the template.

The Wizard creates:

- An Email context with one section in it. The section contains dummy text and one or more **selectors** for variable data, for example: "Hello @first@". You will want to replace those by the names of fields in your data. See "Variable Data" on page 604. The Invoice email template also contains a Dynamic Table; see "Dynamic table" on page 618.
- One **script**, named "To". Double-click that script on the **Scripts** pane to open it. This script ensures that the email is sent to an email address that is specified in a data field called "email-to". After loading data or a data mapping configuration, you can change the script so that it uses the actual field in your data that holds the customer's email address. See "Email header settings" on page 373
- A style sheet, named context_html_email_styles.css, and another style sheet depending on which Template Wizard was used. The style sheets can be found in the **Stylesheets** folder on the **Resources** pane.

The Wizard opens the Email section, so that you can fill it with text and other elements; see "Content elements" on page 465, "Email context" on page 368, and "Email templates" on page 370.

Tip

Use the **Outline** pane at the left to see which elements are present in the template and to

select an element.

Use the **Attributes** pane at the right to see the current element's ID, class and some other properties.

Use the **Styles** pane next to the Attributes pane to see which styles are applied to the currently selected element.

Note that the contents of the email are arranged in tables. The many tables in an Email template ensure that the email looks good on virtually any email client, device and screen size. As the tables have no borders, they are initially invisible.

Tip

Click the **Edges** button on the toolbar to make borders of elements visible on the Design tab. The borders will not be visible on the Preview tab.

Email Template Wizards


There are Wizards for three kinds of Email templates: for **Basic Email**, for **Banded Email**, and **Slate** templates for responsive email designed by Litmus.

Slate: Responsive Email Templates by Litmus

Scroll past the Web Template Wizards to see the Slate: Responsive Email templates, created by Litmus (see <https://litmus.com/resources/free-responsive-email-templates>).

More than 50% of emails are opened on mobile. These five responsive HTML email templates are optimized for small screens and they look great in any inbox. They've been tested in Litmus and are completely bulletproof.

Tip

After creating the email template, click the Responsive Design View icon  at the top of the workspace to see how the email looks on different screen sizes.

The only thing you can set in advance for a Slate template is the color of the call-to-action button. Click the small colored square, right next to the field that holds the default color value, to open the Color dialog and pick a color (see "Color Picker" on page 674). The color can be changed later; see "Colors" on page 583.

Basic Email and Banded Email

The difference between Basic and Banded email is that the contents of a Basic email extend to the email's margin, rather than to the edge of the window in which it is read, as the contents of Banded emails do.

The Banded Email **Action** Template is a simple call-to-action email with a message, header and a button linking to a website, such as an informational or landing page.

The Banded Email **Invoice** Template is an invoice with an optional Welcome message and Pay Now button.

Settings

For a **Blank** email you can not specify any settings in the Wizard.

For an **Action** or **Invoice** email, the Email Template Wizard lets you choose:

- The subject. You can change and personalize the subject later, see "Email header settings" on page 373.
- The text for the header. The header is the colored part at the top. The text can be edited later.
- The color of the header and the color of the button. Click the small colored square, right next to the field that holds the default color value, to open the Color dialog and pick a color (see "Color Picker" on page 674). The color can be changed later; see "Colors" on page 583.
- The web address where the recipient of the email will be taken after clicking the button in the email. Type the URL in the **Link** field.

In addition, for an **Invoice** email you can change the following content settings:

- **Show Welcome Message.** Check this option to insert a salutation and one paragraph with dummy text in the email.

- **Detail Table Name.** Type the name of a detail table to fill the lines of the invoice with data. In the Designer, a detail table is a field in the Data Model that contains a variable number of items (usually transactional data).

Email context

In the Designer the Email context is the folder that contains Email templates. From the Email context, output can be generated in the form of email (see below).

When an Email template is created (see "Creating an Email template with a Wizard" on page 364) or when an Email context is added to a template (see "Adding a context" on page 321) the following happens:

- The Email context is created and one **Email section** is added to it. You can see this on the **Resources** pane: expand the **Contexts** folder, and then expand the **Email** folder. See "Email templates" on page 370 to learn how to fill an Email section. Although only one email can be sent per record when generating Email output, the Email context can contain multiple sections. One Email section is created at the start, but you can add more; see "Adding an Email template" on page 370.
- A style sheet, named context_htmlmail_styles.css, is added to the template. Depending on which Template Wizard was used to create the template, another style sheet can be added as well. Style sheets are located in the folder **Stylesheets** on the **Resources** pane. These style sheets are meant to be used for styles that are only applied to elements in the Email context.

The Wizard opens the Email section, so that you can fill it with text and other elements; see "Content elements" on page 465 and "Email templates" on page 370.

Sending email

When the template is ready, you can generate Email output; See "Generating Email output" on page 973. To test a template, you can send a test email first.

Output, generated from an Email template, can have the following attachments:

- The contents of the Print context, in the form of a single PDF attachment.
- The output of the Web context, as an integral HTML file.
- Other files, an image or a PDF leaflet for example.

Attaching the Print context and/or the Web context is one of the options in the Send (Test) Email dialog.

Note

To split the Print context into multiple attachments, or to attach multiple Web sections as separate attachments, you need to create a Control Script that specifies **parts**; see "Parts: splitting and renaming email attachments" on page 651.

See "Email attachments" on page 379.

Email output settings

The following settings in an Email context influence how the Email output is generated.

Compressing PDF attachments

For PDF attachments, generated from the Print context, you can set the Print Context Image Compression to determine the quality of the files, and with that, the size of the files.

To set the Print Context Image Compression:

1. On the **Resources** pane, expand the **Contexts** folder; then right-click the **Email** context and select **PDF Attachments**.
Alternatively, select **Context > PDF Attachments** on the main menu. This option is only available when editing an Email section in the Workspace.
2. Change the properties of the PDF file that will be attached when the Print context is attached to the email.
Lossless is the maximum quality. Note that this will produce a larger PDF file. Uncheck this option to be able to set a lower quality.
The **quality** is set in a percentage of the maximum quality.
Tile Size is the size of the files in which the image that is being compressed is divided. (If the image height or width is not an even multiple of the tile size, partial tiles are used on the edges.) Image data for each tile is individually compressed and can be individually decompressed. When low Quality values are used to optimize images smaller than 1024 x 1024 pixels, using the largest tile size will produce better results.

Setting a default section for output

When generating output from the Email context, only one of the Email templates can be merged with each record. One of the Email sections is the 'default'; see "Setting a default Email template for output" on page 373.

Email templates

Email templates (also called Email **sections**) are part of the Email context in a template. The Email context outputs HTML email with embedded formatting to an email client through the use of an email server. Since email clients are numerous and do not support same features, the HTML generated by this context is not optimized for any specific client - rather, it's meant to be compatible with as many clients and as many devices as possible.

In Email templates, many content elements can be used; see "Content elements" on page 465. However, special attention must be paid to the way elements are positioned. In Email sections, it is advisable to position elements using Tables and to put text in table cells.

Email templates are personalized just like any other template; see "Variable Data" on page 604.

The subject, recipients (To, CC and BCC), sender and reply-to address are specified with Email Script Wizards; see "Email header settings" on page 373.

An Email context can contain multiple templates. When generating output from the Email context, however, only one of the Email templates can be merged with each record. Set the 'default' Email section (see below) before generating Email output; see also "Generating Email output" on page 973.

Adding an Email template

When an Email template is created (see "Creating an Email template with a Wizard" on page 364), only one Email section is added to it. An Email context may contain various templates, but per record only one of those can be sent when you generate Email output.

It is not possible to add an Email section to an existing Email context with the help of a Template Wizard.

To provide alternative content for your email, you could use Conditional Content (see "Showing content conditionally" on page 613), or Snippets and a script (see "Snippets" on page 548 and "Loading a snippet via a script" on page 640).

If you would like to start with a template that is identical to the one you already have, consider copying it (see "Copying a section" on page 322).

To add a section to the Email context:

- On the **Resources** pane, expand the **Contexts** folder, right-click the **Email** folder, and then click **New Email**.

Deleting an Email template

To delete an Email section:

- On the **Resources** pane, expand the **Contexts** folder, expand the **Email** context, right-click the name of the section, and then click **Delete**.

Warning

No backup files are maintained in the template. The only way to recover a deleted section, is to click **Undo** on the **Edit** menu, until the deleted section is restored. After closing and reopening the template it is no longer possible to restore the deleted context this way.

Styling and formatting an Email template

The contents of an Email section can be formatted directly, or styled with Cascading Style Sheets (CSS). See "Styling and formatting" on page 551.

Email clients do not read CSS files and some even remove a <style> tag when it is present in the email's header. Nevertheless, CSS files can be used with the Email context in the Designer. When generating output from the Email context, the Designer converts all CSS rules that apply to the content of the email to inline style tags, as if local formatting was applied.

Tip

Before you can style an element, you have to select it. In an Email context it can be difficult to select an element by clicking on it. Use the **breadcrumbs** at the top and the **Outline** pane at the left, to select an element. See "Selecting an element" on page 469.

In order for a style sheet to be applied to a specific section, it needs to be included in that section. There are two ways to do this.

Drag & drop a style sheet

1. Click and hold the mouse button on the style sheet on the **Resources** pane.
2. Move the mouse cursor within the **Resources** pane to the section to which the style sheet should be applied.
3. Release the mouse button.

Using the Includes dialog

1. On the Resources pane, right-click the section, then click **Includes**.
2. From the **File types** dropdown, select **Stylesheets**.
3. Choose which CSS files should be applied to this section. The available files are listed at the left. Use the arrow buttons to move the files that should be included to the list at the right.
4. You can also change the order in which the CSS files are read: click one of the included CSS files and use the **Up** and **Down** buttons. Note that moving a style sheet up in the list gives it **less** weight. In case of conflicting rules, style sheets read later will override previous ones.

Note

Style sheets are applied in the order in which they are included in a section. The styles in each following style sheet add up to the styles found in previously read style sheets. When style sheets have a conflicting rule for the same element, class or ID, the **last** style sheet 'wins' and overrides the rule found in the previous style sheet.

Setting a default Email template for output

An Email context can contain multiple templates. When generating output from the Email context, however, only one of the Email templates can be merged with each record.

To select the Email section that will be output by default:

- On the **Resources** pane, expand the **Email** context, right-click a section and click **Set as Default**.

Note

The Default section is executed when the template is merged using the Create Email Content task in Workflow (see Workflow Help: [Create Email Content](#)).

Tip

Use a Control Script to dynamically select an Email section for output depending on the value of a data field.

Email header settings

An email header contains routing information, such as the sender, recipient/s and subject of the message. This topic explains how to make settings for the header of an email that is generated from an Email template.

The default Email **SMTP** settings and the **sender's** name and address are defined in the Connect Designer preferences. They can be adjusted per run in the Send Email and Send Test Email dialogs.

The **subject**, the **recipients** (To, Cc and Bcc), the **sender** and the **Reply to** address can be entered in the **Email Fields** at the top of the workspace. If the fields are not visible, click the words 'Email Fields' (or the small plus before them) to expand the Email Fields area.

To use a variable email address in any of the fields, simply **drag and drop** a data field into the email field.

The specified subject and addresses will be visible when viewing the email in the workspace in

Preview mode.

The To address must always be variable. This field is not used when you send a test email (see "Generating Email output" on page 973).


Note

Using a variable email address requires you to load data or a data mapping configuration first; see "Loading data" on page 594.

Using the Email Script Wizard

In addition to the drag and drop method, you can use the Email Script Wizard to add data to an email header field. It lets you choose one or more data fields and enter a prefix and/or suffix (per data field).

There are two ways to open the Email Script Wizard:

- Via the **Email Fields**. Open the email section and expand the Email Fields at the top by clicking **Email Fields**. Click the word before the email field that you want to set. If there already is a script for that field, that script will be opened. Otherwise, a new script will be created and opened.
- Via the **Scripts** pane. Click the black triangle on the **New** button  and select the respective email script. A new script will be added to the Scripts pane. Double-click the new script to open it.

The default script adds the content of the selected data field to the header field.

If you want to write a more complex script, click the **Expand** button. The result of the script should be a valid, fully-formed email address.

The language in which the script has to be written is JavaScript. For more information on writing scripts, see "Writing your own scripts" on page 624.

Other header fields

At some point you may need to define a header field that isn't available in the Preferences or in the Email Fields. This can be done in a Control Script. For a few examples of such scripts, see "Adding custom ESP handling instructions" on page 978. To get started with Control Scripts, refer to "Control Scripts" on page 645.

Email SMTP settings

Simple Mail Transfer Protocol (SMTP) is the standard protocol for sending emails across the Internet.

Default SMTP settings can be specified in the Preferences dialog: select **Window > Preferences**, expand the **Email** preferences and click **SMTP**.

You can add as many presets as needed, for example for different Email Service Providers (see "Using an ESP with PlanetPress Connect" on page 976). To do this, click the Add button at the right. Then fill out the following settings:

- **Name:** The name of the preset. This will show up in the Send Email dialog.
- **Host:** The SMTP server through which the emails are to be sent. This can be a host (mail.domain.com) or an IP address.
- **Port:** You can specify a port number. This will be added to the host name, for example: smtp.mandrillapp.com:465.
- **Use authentication:** Check this option and fill in the user name if a user name and password are needed to send emails through the host. (The password has to be specified in the Send Email or Send Test Email dialog.)
- **Start TLS:** This option is enabled if authentication is checked. It sends emails through Transport Layer Security (TLS), which is sometimes referred to as SSL.

When you click the **Restore** button, the presets for a number of Email Service Providers will appear.

Note

When updating the software from a version prior to version 1.5, pre-existing presets will be maintained in the new version.

In the Send Test Email dialog and Send Email dialog (see "Send (Test) Email" on page 723) you will be able to choose one of the presets and adjust the settings to your needs.

Subject

To specify a subject for an email template:

1. Open the email section and expand the Email Fields by clicking **Email Fields** at the top of the section.
2. Type the subject in the **Subject** field.

To add **variable data** to the subject of an email section, **drag and drop** a data field into the Subject field at the top of the workspace. Two things will happen:

- A placeholder for the data field appears in the subject line (for example: @email@).
- A new script, named Subject, is added to the Scripts pane.

You can add as many data fields to the subject as you like. When you do add more than one data field, the existing Subject script will be modified to include all data fields that are added to the subject.

The result of the script will be visible in the Subject field in Preview mode: click the Preview tab at the bottom of the workspace.

Note

By default, the Subject script targets one email section specifically. You can see this when you double-click the script on the Scripts pane. The selector of the Subject script contains the name of a particular email section, for example: `html[section="Content"]` (in this case, Content is the name of the email section). If you remove the `html[...]` part from the selector, the script will work for all email sections.

Subject scripts made with earlier versions of the software are **not** specific to one email section.

Writing a custom Subject script

The default script replaces all @field@ placeholders in the subject line with field values. This script can be modified, for example to create a subject that depends on the value of a data field. Open the Script Wizard (see "Using the Email Script Wizard" on page 374), click the **Expand** button and modify the script.

If you don't know how to write a script, see "Writing your own scripts" on page 624 first.

Note

A Subject script created by clicking Subject in the Email Fields always targets one email section specifically. Remove the `html [. . .]` part from the **selector** of the script to make the script work for all email sections.

Recipients: To, CC and BCC

To specify recipients for Email output, you can simply **drag and drop** a data field that contains an email address into the **To** field at the top of the workspace. A new script, named To, will be added to the Scripts pane.

Note that you can add only one data field to the email field this way. When you drag another data field into the email field the existing script will be replaced..

Email addresses can be added to the **Cc** and **Bcc** fields in the same manner, but it is also possible to type an email address directly in the Cc or Bcc field (as long as no script is present for that field).

Email addresses in the Bcc ('blind carbon copy') field will not be visible to any other recipient of the email.

Alternatively, you could use the Script Wizard to create the scripts; see "Using the Email Script Wizard" on page 374.

Sender

From address

A default **From** name and email address can be specified in the Preferences dialog: select **Window > Preferences**, expand the **Email** preferences and click **General**.

This name and email address will appear as the default in the Send Email and Send Test Email dialogs (see "Send (Test) Email" on page 723).

The default can be overwritten by typing an email address directly in the From field (as long as no script is present for this field).

Using the Script Wizard you can create a **dynamical** From address; see "Using the Email Script Wizard" on page 374. It is also possible to drag and drop one data field into the From field directly.

Tip

A dynamical From address is often used when sending email campaigns and to do tracking of email replies. Include the recipient's email address in a dynamic From address to enable automatic detection and removal of undeliverable e-mail addresses. (This technique is called VERP; see [Wikipedia](#).)

Note

When sending emails with a variable From address through PlanetPress Workflow, check the option **Precedence to template address** in the Create Email Content task properties to make sure that the dynamic address gets precedence over the email address specified in the task properties (see [Create Email Content task](#)).

Reply To address

The Reply To address is used by mail clients, when the recipient clicks the Reply To (or Reply All) button.

You can type an email address directly in the Reply To field (as long as no script is present for this field).

Alternatively, you can drag and drop one data field into the field, or use the Email Script wizard (see "Using the Email Script Wizard" on page 374), to specify the Reply To address in a script.

Email PDF password

The Email PDF Password Script Wizard defines a password with which to protect the PDF generated when using the Print context as PDF Attachment option in the Send Email or Send Test Email dialogs (see "Generating Email output" on page 973). The result of the script will be the password necessary to open the PDF when it is received by email.

To define a password to protect the generated PDF attachment:

1. On the Scripts pane, click the black triangle on the **New** button and click **Email PDF password Script**. A new script is added to the Scripts pane.
2. Double-click the new script to open it.
3. Select a data field and optionally, type a prefix and/or suffix.

Password types

PDF allows for two types of passwords to be set on a secured PDF file: a user password and owner password. The user password allows a limited access to the file (e.g. printing or copying text from the PDF is not allowed). The owner password allows normal access to the file. The Email PDF password script sets both the user and owner password to the same value, so that when the recipient provides the password, he can manipulate the file without limitations.

Note

If a template has a Control Script that creates multiple PDF attachments, all the attachments are secured by the same password.

Note

Via a Control Script it is possible to set a different user password and owner password, see "Control Script: Securing PDF attachments" on page 658, "Control Scripts" on page 645 and "Control Script API" on page 930.

Email attachments

Output, generated from an Email template, can have the following attachments:

- The contents of the Print context, in the form of a single PDF attachment.
- The output of the Web context, as an integral HTML file.
- Other files, an image or a PDF leaflet for example.

Attaching the Print context and/or the Web context is one of the options in the Send (Test) Email dialog.

By default, when adding the Print context to an email, all Print sections are output to a single PDF file, named after the email subject, which is then attached to the email. The PDF can be protected with a password (see "Email PDF password" on the previous page).

When adding the Web context to an email, only the default Web section is generated and added to the email as an HTML file that is named after the email subject.

Note

To split the Print context into multiple attachments, or to attach multiple Web sections as separate attachments, you need to create a Control Script that specifies **parts**; see "Parts: splitting and renaming email attachments" on page 651.

This topic explains how to attach files other than those generated by the Print or Web context.

Attaching external files

To attach files other than those generated by the Print or Web context to Email output:

1. Add the files to the template (see "Adding images" on page 539) or put them in a folder that is available to the machine that outputs the emails.
2. Create a script: on the **Scripts** pane at the bottom left, click **New**. A new script appears in the list. Double-click on it to open it. If you are not familiar with scripts, see "Writing your own scripts" on page 624 for an explanation of how scripts work.
3. Change the name of the script, so that it reflects what the script does.
4. Choose the option **Selector** and in the **Selector** field, type **head**.
5. Write a script that appends a `<link>` element to the `results` (the selector is `head`, so the `results` contain the `<head>` of the email). For example:

```
results.append("<link rel='related' href='images/letter-  
CU00048376.pdf'>");
```

- Make sure to set the **rel** attribute to **related**.
- The **href** attribute determines where the file comes from.
For resources inside the template, use `'images/file.extension'`, or `'fonts/myfont.otf'`, etc.
For external resources, you need the full path to the file, for example:
`'file:///c:/resources/attachments/instructions.pdf'`
or, for a remote file:
`'http://localhost:8080/pod/v1/deliverynotes/{8FCEC8BC-72E8-486B-A206-516BF10E21F6}'`.
Of course, you can also use dynamic calls such as
`'file:///c:/clientfiles/' + record.fields.client_id +
'/invoices/' + record.fields.invoice_number + '.pdf'`.
- Add the **title** attribute to specify a custom attachment name. For example:

```
results.append("<link rel='related' href='images/{8FCEC8BC-
```



```
72E8-486B-A206-516BF10E21F6}.pdf'  
title='INV1375461.pdf'>");
```

Examples

The following script attaches a PDF file named **letter-CU00048376.pdf** to each generated email. The PDF file is located in the Images folder on the Resources panel.

```
results.append("<link rel='related' href='images/letter-  
CU00048376.pdf'>");
```

If that same file would be located on the C: drive, the script should refer to it as follows:

```
href='file:///C:/letter-CU00048376.pdf'.
```

The link doesn't have to be static; you could use data from the record set to build the link, for example:

```
var customerID = record.fields.ID;  
results.append('<link rel="related" href="images/letter-' +  
customerID + '.pdf">');
```

Renaming attachments

External files that are sent as attachments can be renamed via the script that attaches them to the email, by putting their intended name in the **title** attribute. For example:

```
results.append("<link rel='related' href='images/{8FCEC8BC-72E8-486B-A206-  
516BF10E21F6}.pdf' title='INV1375461.pdf'>");
```

Print and **Web** sections that are attached to an email can be renamed via a Control Script; see "Parts: splitting and renaming email attachments" on page 651.

Web

With the Designer you can create one or more Web templates and merge the template with a data set to generate personal web pages.

The **Web** context is the Web output channel and the folder in the Designer that can contain one or more Web templates. CaptureOnTheGo templates are a special kind of Web templates. They are stored in the Web folder as well.

The Web context outputs one HTML web page that contains the HTML text and all the resources necessary to display it. JavaScript files are added to the <head> in the generated HTML file. They are useful to add special features such as those offered by jQuery and its plugins, or MooTools. Style sheets are also added to the <head> and are used just as they would be used in a regular web page.

It is advisable to follow design guidelines for web pages, so that they are likely to look good in different browsers and on different devices and screen sizes. When you start with a Web Template Wizard, the Foundation framework is added to the template, to guarantee just that; see "Creating a Web template with a Wizard" below and "Capture OnTheGo template wizards" on page 416.

When a Web template is created, either with a Wizard or by adding the Web context to an existing template (see "Adding a context" on page 321), the Web context folder is created along with other files that are specific to an Web context; see "Web Context" on page 386.

Many of the content elements that are available for all three contexts are particularly suitable for web pages; see "Content elements" on page 465. Web templates are personalized just like any other template; see "Variable Data" on page 604.

Only one Web section is created at the start, but you can add as many Web sections as you need; see "Web pages" on page 387. Note that when the Designer merges a data set to generate output from the Web context, it can merge only one of the templates with each record; see "Generating Web output" on page 981.

Creating a Web template with a Wizard

With the Designer you can design Web templates and output them through Workflow or as an attachment to an email when generating Email output.

Capture On The Go templates are a special kind of Web templates; see "Capture OnTheGo template wizards" on page 416.

A Web Template Wizard helps you create a Web page that looks good on virtually any browser, device and screen size.

Foundation

All Web Template Wizards in Connect Designer make use of the Zurb **Foundation** front-end framework. A front-end framework is a collection of HTML, CSS, and JavaScript files to build upon. Foundation is a **responsive** framework: it uses CSS media queries and a mobile-first approach, so that websites built upon Foundation look good and function well on multiple devices including desktop and laptop computers, tablets, and mobile phones. Foundation is tested across many browsers and devices, and works back as far as IE9 and Android 2. See <http://foundation.zurb.com/learn/about.html>.

For more information about the use of Foundation in the Designer, see "Using Foundation" on page 420.

After creating a Web template, the other contexts can be added, as well as other sections (see "Adding a context" on page 321 and "Adding a Web page" on page 388).

To create a Web template with a Template Wizard:

1.
 - In the **Welcome** screen that appears after startup, choose **Browse Template Wizards**.
Scroll down until you see the **Foundation Web Page Starter** Template Wizards.
 - Alternatively, on the **File** menu, click **New**, expand the **Template** folder, and then expand the **Foundation Web Page Starter** folder.
2. Select a template. There are 4 types of Web Template Wizards :
 - Blank
 - Contact Us
 - Jumbotron
 - Thank You

If you don't know what template to choose, see "Web Template Wizards" on page 385 further down in this topic, where the characteristics of each kind of template are described.

3. Click **Next** and make adjustments to the initial settings.
 - **Section:**
 - **Name:** Enter the name of the Section in the Web context. This has no effect on output.

- **Description:** Enter the description of the page. This is the contents of a <meta name="description"> HTML tag.
- **Top bar** group:
 - **Set width to Grid:** Check this option to limit the width of the top bar contents to the Foundation Grid, instead of using the full width of the page.
 - **Stick to the top of the browser window:** Check to lock the top menu bar to the top of the page, even if the page has scroll bars. This means the menu bar will always be visible in the browser.
 - **Background color:** Enter a valid hexadecimal color code for the page background color (see [w3school's color picker](#)) , or click the colored circle to the right to open the Color Picker.
- **Colors** group: Enter a valid hexadecimal color code (see [w3school's color picker](#)) or click the colored square to open the Color Picker dialog (see "Color Picker" on page 674), and pick a color for the following elements:
 - **Primary:** links on the page.
 - **Secondary:** secondary links on the page.
 - **Text:** text on the page contained in paragraphs (<p>).
 - **Headings:** all headings (<h1> through <h6>) including the heading section's subhead.

4. Click **Finish** to create the template.

The Wizard creates:

- A Web context with one web page template (also called a **section**) in it. The web page contains a Header, a Section and a Footer element with dummy text, and depending on the type of web page, a navigation bar, button and/or Form elements.
- Resources related to the Foundation framework (see "Web Template Wizards" on the next page): style sheets and JavaScript files. The style sheets can be found in the **Stylesheets** folder on the **Resources** pane. The JavaScript files are located in the **JavaScript** folder on the **Resources** pane, in a **Foundation** folder.
- A collection of Snippets in the **Snippets** folder on the Resources pane. The Snippets contain ready-to-use parts to build the web page. Double-click to open them. See "Snippets" on page 548 for information about using Snippets.
- Images: icons, one picture and one thumbnail picture. Hover your mouse over the names of the images in the **Images** folder on the **Resources** pane to get a preview.

The Wizard opens the Web section, so that you can fill it with text and other elements; see "Content elements" on page 465, "Web Context" on the facing page and "Web pages" on page 387.

Web pages can be personalized just like any other type of template; see "Variable Data" on page 604 and "Personalizing Content" on page 592.

Tip

Use the **Outline** pane at the left to see which elements are present in the template and to select an element.

Use the **Attributes** pane at the right to see the current element's ID, class and some other properties.

Use the **Styles** pane next to the Attributes pane to see which styles are applied to the currently selected element.

Tip

Click the **Edges** button on the toolbar to make borders of elements visible on the Design tab. The borders will not be visible on the Preview tab.

Web Template Wizards

Foundation

All Web Template Wizards in Connect Designer make use of the Zurb **Foundation** front-end framework. A front-end framework is a collection of HTML, CSS, and JavaScript files to build upon. Foundation is a **responsive** framework: it uses CSS media queries and a mobile-first approach, so that websites built upon Foundation look good and function well on multiple devices including desktop and laptop computers, tablets, and mobile phones. Foundation is tested across many browsers and devices, and works back as far as IE9 and Android 2. See <http://foundation.zurb.com/learn/about.html>.

Jumbotron

The name of the Jumbotron template is derived from the large screens in sports stadiums. It is most useful for informative or marketing-based websites. Its large banner at the top can display

important text and its "call to action" button invites a visitor to click on to more information or an order form.

Contact Us

The Contact Us template is a contact form that can be used on a website to receive user feedback or requests. It's great to use in conjunction with the Thank You template, which can recap the form information and thank the user for feedback.

Thank You

The Thank You template displays a thank you message with some text and media links.

Blank web page

The Blank Web Page template is a very simple Foundation template that contains a top bar menu and some basic contents to get you started.

Web Context

In the Designer the Web context is the folder that contains Web page templates.

Creating the Web context

You can start creating a Web template with a Wizard (see "Creating a Web template with a Wizard" on page 382), or add the Web context to an existing template (see "Adding a context" on page 321).

When a Web template is created the following happens:

- The Web context is created and one Web page or **section** is added to it. You can see this on the **Resources** pane: expand the **Contexts** folder, and then expand the **Web** folder. See "Web pages" on the next page to learn how to fill a web page template in the Designer.
Although only one web page can be generated per record when generating Web output, the Web context can contain multiple sections. One section is created at the start, but you can add more; see "Adding a Web page" on page 388.

- A style sheet, named `context_web_styles.css`, is added to the template. If a Template Wizard was used to create the template, Foundation style sheets are added as well. Style sheets are located in the folder **Stylesheets** on the **Resources** pane. These style sheets are meant to be used for styles that are only applied to elements in the Web context; see "Styling and formatting" on page 551.

Generating Web output

When the template is ready, you can:

- Output the web page as an as an integral HTML file attached to an Email context in the same template.
- Output the Web context in an automated Workflow using the Create Web Content task (see Workflow Help: [Create Web Content](#)).

See "Generating Web output" on page 981.

Includes

The Web context outputs one HTML web page that contains the HTML text and all the resources necessary to display it. JavaScript files are added to the `<head>` in the generated HTML file. They are useful to add special features such as those offered by jQuery and its plugins, or MooTools. Style sheets are also added to the `<head>` and are used just as they would be used in a regular web page.

Which style sheets and JavaScript files are included, and in which order, can be decided for the **Web context** as a whole, as well as per Web section.

To make this setting for the Web context as a whole, right-click the context on the **Resources** pane and select **Includes**. Alternatively, select **Context > Includes** on the main menu. (Note that this option is only available when editing a Web section in the Workspace.) For further explanation see: "Includes dialog" on page 682.

To make this setting for a particular Web section, right-click the section on the **Resources** pane and select **Includes**.

Web pages

Web pages (also called Web **sections**) are part of the Web context (see "Web Context" on the previous page) in a template.

The Web context outputs one HTML web page that contains the HTML text and all the resources necessary to display it. JavaScript files are added to the <head> in the generated HTML file. They are useful to add special features such as those offered by jQuery and its plugins, or MooTools. Style sheets are also added to the <head> and are used just as they would be used in a regular web page.

A Web context can contain multiple templates. When generating output from the Web context, however, only one of the Web templates can be merged with each record. Set the 'default' Web section (see "Setting a default Web page for output" on page 391) before generating Web output; also see "Generating Web output" on page 981.

Creating a Web page

When creating a Web page, it is advisable to follow design guidelines for web pages, so that they are likely to look good in different browsers and on different devices and screen sizes. When you start with a Web Template Wizard, the Foundation framework is added to the template, to guarantee just that; see "Creating a Web template with a Wizard" on page 382. Other approaches are described below, in "Adding a Web page" below.

Adding a Web page

When a Web template is created (see "Creating a Web template with a Wizard" on page 382), only one Web section is added to it. A Web context may contain various templates, but per record only one of those can be used to generate output.

It is not possible to add a Web section to an existing Web context with the help of a Template Wizard.

To provide alternative content for the web page, you could use Conditional Content (see "Showing content conditionally" on page 613), or Snippets and a script (see the Help topics "Snippets" on page 548 and "Loading a snippet via a script" on page 640, and this how-to: [Multi-page Web template.](#))

Tip

For an example of how to serve different web pages using snippets, see the following how-to: [Creating a multi-page Web template.](#)

If you would like to start with a template that is identical to the one you already have, consider copying it (see "Copying a section" on page 322).

To add a blank section to the Web context:

- On the **Resources** pane, expand the **Contexts** folder, right-click the **Web** folder, and then click **New Web page**.

Deleting a Web page

To delete a Web section:

- On the **Resources** pane, expand the **Contexts** folder, expand the **Web** context, right-click the name of the section, and then click **Delete**.

Warning

No backup files are maintained in the template. The only way to recover a deleted section, is to click **Undo** on the **Edit** menu, until the deleted section is restored. After closing and reopening the template it is no longer possible to restore the deleted context this way.

Filling a Web page

Many of the content elements that are available for all three contexts are particularly suitable for web pages; see "Content elements" on page 465. Do not use Positioned Boxes and Tables to position elements, however; use Inline Boxes instead.

Forms and Form elements are only available in a Web context; see "Forms" on page 527 and "Form Elements" on page 532.

Using variable data on a Web page

Web templates are personalized just like any other template; see "Variable Data" on page 604. There are a few extra possibilities, though: variable data can be used in Form elements and they can be passed to client-side JavaScript.

Using Variable Data in Form elements

Variable data may be used in form elements, such as a drop-down list (a Select element). How to do that, is described in this how-to: [Dynamically add options to a dropdown](#).

Passing Variable Data to client-side JavaScript

When serving Web pages using Workflow, the HTML is first personalized and then served to the web browser by a Workflow process. At that stage custom JavaScripts do not have access to the information stored in the Data Model. To enable a client-side script to use variable data, you need to create a Text Script that produces a JSON string and stores that in the attribute of an HTML element, the `value` attribute of a hidden field for example. The custom JavaScript can then retrieve that information and use it to create dynamic page elements. Producing a JSON string and storing the results in the attribute of an HTML element are both options in the Text Script wizard; see "Using the Text Script Wizard" on page 607.

Styling and formatting a Web page

The contents of a Web section can be formatted directly, or styled with Cascading Style Sheets (CSS). See "Styling and formatting" on page 551.

In order for a style sheet to be applied to a specific section, it needs to be included in that section. There are two ways to do this.

Drag & drop a style sheet

1. Click and hold the mouse button on the style sheet on the **Resources** pane.
2. Move the mouse cursor within the **Resources** pane to the section to which the style sheet should be applied.
3. Release the mouse button.

Using the Includes dialog

1. On the Resources pane, right-click the section, then click **Includes**.
2. From the **File types** dropdown, select **Stylesheets**.
3. Choose which CSS files should be applied to this section. The available files are listed at the left. Use the arrow buttons to move the files that should be included to the list at the right.

4. You can also change the order in which the CSS files are read: click one of the included CSS files and use the **Up** and **Down** buttons. Note that moving a style sheet up in the list gives it **less** weight. In case of conflicting rules, style sheets read later will override previous ones.

Note

Style sheets are applied in the order in which they are included in a section. The styles in each following style sheet add up to the styles found in previously read style sheets. When style sheets have a conflicting rule for the same element, class or ID, the **last** style sheet ‘wins’ and overrides the rule found in the previous style sheet.

Note

Which style sheets are included can also be set for the **Web context** as a whole: in step 1, right-click the Web context, instead of a section.

Setting a default Web page for output

When generating output from the Web context, only one of the Web templates can be merged with each record.

To select the Web section that will be output by default:

- On the **Resources** pane, expand the **Web** context, right-click a section and click **Set as Default**.

Tip

Use a Control Script to dynamically select a Web section for output depending on the value of a data field. See "Control Scripts" on page 645.

Including JavaScript files

Which JavaScript files are included in the a Web section, depends on a setting for that section. To change this:

1. On the **Resources** pane, right-click a section in the **Web** context and click **Includes**.
2. From the **File types** dropdown, select **JavaScripts**.
3. Choose which JavaScript files should be included in this section. The list at the left displays the JavaScript files that are present in the template's resources. The list at the right shows the style sheets and or JavaScript files that will be included in the output of the current section (or Web sections, if you have selected the Web context). Use the **Include** and **Exclude** buttons to move files from one list to the other.
4. Click one of the included files and use the **Up** and **Down** buttons to change the order in which the files are included.

For more information about using JavaScript files, see "Using JavaScript" on page 403.

Setting the title, meta data and a shortcut icon

Each Web section has a set of properties to define the title of the web page, the shortcut icon and the meta tags appearing in the web page's head (with the HTML tag: <head>, see http://www.w3schools.com/tags/tag_head.asp).

To change these properties:

1. On the **Resources** pane, expand the **Web** context, right-click the section and click **Properties**.
2. Enter the **Page Title**. The contents of this field will go in the <title> HTML tag. (**Name** is the name of the section in the Web context; this has no effect on output.)
3. Add a **Shortcut Icon** by entering the path to the favicon.ico file, for instance `images/favicon.ico`.

Tip

If a valid favicon image is dragged to the Web section, it will automatically be set as a shortcut icon.

4. The **Meta Information Group** lists all <meta> tags that will be added to the header of the HTML file generated in the output. Click the **Add** button to add a new <meta> tag to the list. Then you can select the type of <meta> tag, which is either name or http-equiv, and enter the value (for a name-type meta tag) or the content. For more information on <meta>

tags, see [W3Schools - HTML meta tag](#).

Adding information to the <head> via script

When generating Web output, the Designer automatically adds the included resources to the <head>. To add other tags to the <head>, such as a <base> tag to set a default base URL/target for all relative URLs in a document, you need to write a script. If you are not familiar with scripts, see "Writing your own scripts" on page 624 for an explanation of how scripts work.

1. Create a script: on the **Scripts** pane at the bottom left, click **New**. A new script appears in the list. Double-click on it to open it.
2. Change the name of the script, so that it reflects what the script does.
3. Choose the option **Selector** and in the **Selector** field, type `head`.
4. Write a script that appends an element to the <head> of the web page.

Example

The following script adds a <base> element to the head of a web page.

```
results.append("<base href='http://www.w3schools.com/images/'  
target='_blank'>");
```

Forms

Web templates can contain Forms. Capture OnTheGo templates always contain a Form.

Tip

To create a Capture OnTheGo template, preferably use a Template Wizard (see "Capture OnTheGo template wizards" on page 416). The Wizard doesn't just add the form, it also adds the necessary Capture OnTheGo form elements (see), style sheets and JavaScript files, and extra pre-made elements.

Adding a Form

This procedure describes how to add a Form element to an existing Web context.

1. On the **Resources** pane, expand the **Web** context and double-click a Web page to open it.
2. To use the Form Wizard, select the **Insert > Form Wizard** menu option. The Form Wizard adds a Form to the Web page including the specified fields.
Alternatively, you can select **Insert > Form** on the menu to open a dialog that lets you set the Form's properties, validation method and location, but doesn't allow you to specify fields. If you choose this method, skip step 8 and 9 of this procedure and add fields after inserting the Form (see "Adding elements to a Form" on page 398).
3. Add an **ID** and/or a **class**. ID's and classes are particularly useful with regard to variable data (see "Personalizing Content" on page 592) and styling (see "Styling templates with CSS files" on page 553).
4. In the **Action** field, enter the URL where the form data should be sent. The URL should be a server-side script that can accept form data. The **action** of a Capture OnTheGo form should specify the Workflow HTTP Server Input task that receives and handles the submitted data. The action will look like this: **http://127.0.0.1:8080/action** (8080 is Workflow's default port number; 'action' should be replaced by the HTTP action of that particular HTTP Server Input task). The **method** of a Capture OnTheGo form should be **POST** to ensure that it doesn't hit a data limit when submitting the form. The GET method adds the data to the URL, and the length of a URL is limited to 2048 characters. Especially forms containing one or more Camera inputs may produce a voluminous data stream that doesn't fit in the URL. GET also leaves data trails in log files, which raises privacy concerns. Therefore POST is the preferred method to use.
5. Using the the **Method** drop-down, select whether the form should be sent using the GET or POST method.
6. Using the next drop-down, select the form's Encryption Type (**enctype**):
 - **application/x-www-form-urlencoded**: Default. All characters are encoded before they are sent. Spaces are converted to "+" symbols, and special characters are converted to ASCII HEX values.
 - **multipart/form-data**: No characters are encoded. This value is required when you are using forms that have a file upload control.
 - **text/plain**: Spaces are converted to "+" symbols, but no special characters are encoded.

7. Select a **validation** method:

- The **Browser** validation method leaves it up to the browser to validate the user input. When adding fields to the Form (see the next step) you can only make fields required and set the maximum length as an additional requirement for some fields.
- Select **JQuery Validation** to validate using JQuery scripts. This allows you to specify stricter requirements per field and type a different message for each field to display to the user if the input is not valid. This method ensures a more consistent validation as it is browser independent. The necessary JQuery files will be added to the JavaScript folder on the Resources pane when the form is inserted.

8. Under **Fields**, click the **Add** button and click on the desired field type to add a field of that type; see "Form Elements" on page 532.

Note

A Fieldset is not available in the Form Wizard, because a Fieldset itself can contain multiple different fields. Add the Fieldset after inserting the Form; see "Adding elements to a Form" on page 398.

9. Double-click each field in the Fields list and change its settings. For an explanation of the settings, see "Adding elements to a Form" on page 398.

10. The order of the elements in the list under **Fields** determines in which order the elements will be added to the Form. Use the **Move Up** and **Move Down** buttons to change the order of the elements in the list.

11. Use the **Location** drop-down to select where to insert the element.

- **At cursor position** inserts it where the cursor is located in the template.
- **Before element** inserts it before the HTML element in which the cursor is currently located. For example if the cursor is within a paragraph, the insertion point will be before the <p> tag.*
- **After start tag** inserts it within the current HTML element, at the beginning, just after the start tag.*
- **Before end tag** inserts it within the current HTML element, at the end, just before the end tag.*

- **After element** inserts it after the element in which the cursor is currently located. For example if the cursor is within a paragraph, the insertion point will be after the end tag of the paragraph (</p>).*

* If the current element is located inside another element, use the **Elements** drop-down to select which element is used for the insertion location. The list displays every element in the breadcrumbs, from the current selection point until the root of the body.

12. Close the dialog. Now you can start adding elements to the Form (see "Using Form elements" on page 398, "Form Elements" on page 532, and "COTG Elements" on page 519).

Changing a Form's properties and validation method

Once a Form has been added, you can of course edit its HTML code directly in the Source view of the workspace. Apart from that, there are a number of dialogs to change a Form's properties and validation settings.

Changing a Form's properties

1. Select the form (see "Selecting an element" on page 469).
2. On the **Attributes** pane you can change:
 - The **ID** and/or **class**. ID's and classes are particularly useful with regard to variable data (see "Personalizing Content" on page 592) and styling (see "Styling templates with CSS files" on page 553).
 - An **Action**: the URL where the form data should be sent. The URL should be a server-side script that can accept form data.
 - A **Method**: this defines whether the form should be sent using the GET or POST method.
 - An Encryption Type (**enctype**):
 - **application/x-www-form-urlencoded**: Default. All characters are encoded before they are sent. Spaces are converted to "+" symbols, and special characters are converted to ASCII HEX values.
 - **multipart/form-data**: No characters are encoded. This value is required when you are using forms that have a file upload control.
 - **text/plain**: Spaces are converted to "+" symbols, but no special characters are encoded.

Changing a Form's validation method

In Connect PlanetPress Connect, there are two ways in which a Form's input can be validated:

- The **Browser** validation method leaves it up to the browser to validate the user input. When adding fields to the Form (see the next step) you can only make fields required and set the maximum length as an additional requirement for some fields.
- **JQuery Validation** validates input using JQuery scripts. This allows for stricter requirements per field and a different message for each field to display to the user if the input is not valid. This method ensures a more consistent validation, as it is browser independent. The necessary JQuery files will be added to the JavaScript folder on the Resources pane when this option is chosen.

To change a Form's validation method:

1. **Right-click** on the Form element and choose **Validation settings**.
2. Choose a validation type (see above).
3. Double-click each field in the list to edit their validation settings:
 - **Required**: Check if the field is required to submit the form. If a field is required but contains no data, a message will be shown to the user.
 - **Minimum length**: Enter a numerical value for the minimum character length required for this field.
 - **Maximum length**: Enter a numerical value for the minimum character length accepted for this field.
 - **Equal to**: Use the drop-down to select another field that is already added to the same Form. The contents of both fields must match for the data to be validated. This is useful for confirmation fields such as for passwords, email addresses etc.

Which of these options are available depends on the validation method of the form: with Browser validation you can only make a field required and set a maximum length.

Changing a Form's validation in HTML

In HTML, the validation method is stored in the `data-validation-method` attribute of the `<form>` element, with the value "browser" or "jquery-validation".

A custom message to be shown when validation of a particular Form element has failed, can be stored in the `data-custom-message` attribute of the Form element, for example:

```
<input id="email1" name="email1" data-custom-message="Enter a valid email address." type="email" required="">
```

Validation in Connect 1.0.0

In Connect 1.0.0, the validation method of the template was stored using the names "standard" and "custom". Standard has changed to "browser" and custom is now "jquery-validation". When you open a template made with that version of the software, the template will be migrated to use the new attribute values for the `data-validation-method` attribute of the `<form>` element. The JavaScript file `web-form-validation.js` will not be migrated: delete that file and then change the Form's validation method to jQuery Validation, as described above. When you click OK, the new version of the `web-form-validation.js` file will be added.

Submitting a Form

When a form is submitted, by clicking or touching the Submit button, the **name** and **value** of form elements are sent to the address that is specified in the Form's action (see "Adding a Form" on page 393 or "Changing a Form's properties" on page 396). If the `name` attribute is omitted, the data of that input field will not be sent at all.

The Form's validation should ensure that the data that the user submits is valid.

Using Form elements

Web Form elements can be used in a Web Form or in a Capture OnTheGo Form (see "Forms" on page 527 and "Capture OnTheGo" on page 406). This topic explains how to add these elements to a Form and how to prepare them so that when the Form is submitted, they provide valid data that can be handled easily.

For a list of Form elements, see "Form Elements" on page 532.

For a list of the extra elements that can be used in a Capture OnTheGo form, see "COTG Elements" on page 519.

Adding elements to a Form

To add an element to a Form or Fieldset, click inside the Form or Fieldset, select **Insert > Form elements**, and choose the respective element on the menu. (When the element isn't available via the menu, see the tip below.) Now you can change the element's settings:

1. Add an **ID** (required) and, optionally, a **class**.

Note

The ID will be copied to the `name` attribute of the element. The `name` attribute is what identifies the field to the receiving server-side script. To change the name, select the element after inserting it and type the new name on the **Attributes** pane.

ID's and classes are also useful with regard to variable data (see "Personalizing Content" on page 592) and styling (see "Styling templates with CSS files" on page 553).

2. Type a label, or choose No label under Style, to omit the label. (For Label elements there are no other options to be set.)
3. If applicable, choose a style for the label (for the label of a Checkbox, for example, you can't set a style).
 - **Wrap input with label** places the input element inside the Label element.
 - **Attach label to input** ties the label to the input element using the `for` attribute of the Label element.
 - **Use label as placeholder** inserts the given label text in the placeholder attribute of the field.
 - **No style** omits the label altogether.

Note

The first two label styles ensure that when the user clicks the label, the input element gets the focus.

4. The following options are only available for specific elements:
 - For a **Text Area** you can specify a **number of rows**.
 - For a **Radio Button**, the **submit name** indicates to which Radio Button Group the Radio Button belongs.
 - For a **Button**, **Checkbox**, **Hidden Field**, and **Radio Button** you can set the **value**. The value is associated with the input and will be sent on submitting the Form.

Tip

For other Form elements, you can set the **default value** to be the value of a field in the record set; see "Specifying a default value" on the next page.

- For a **Checkbox** or **Radio Button** you can check **checked** or **selected** respectively for the element to initially be checked/selected when the web page is shown.
 - For a **Button**, you can set the **button type**:
 - **Submit**: The button will validate the form data and if validation is successful, send the data to the provided URL (the action specified for the Form; see "Changing a Form's properties" on page 530).
 - **Reset**: The button will reset the form to its original configuration, erasing any information entered and options provided. **Note**: This cannot be undone!
5. Depending on the validation method of the form (see "Changing a Form's validation method" on page 531) and the type of element there are a number of options to set under **Validation**. With Browser validation you can only make a field required and set a maximum length.
- **Required**: Check if the field is required to submit the form. If a field is required but contains no data, a message will be shown to the user.
 - **Minimum and maximum length**: Enter a numerical value for the minimum and maximum character length required for this field.
 - **Equal to**: Use the drop-down to select another field that is already added to the same Form. The contents of both fields must match for the data to be validated. This is useful for confirmation fields such as for passwords, email addresses etc.
6. Under **Warnings**, type the message that will be displayed to the user if the input is not valid.

The **name** attribute of Form elements is sent to the server (together with the input value) after the form has been submitted. When adding an element to a Form or Fieldset, you cannot specify a `name`; the ID will be copied to the element's `name` attribute. After adding the element to the Form or Fieldset you can change the `name` on the Attributes pane.

Adding new HTML5 elements

HTML5 added several new input element types that can't be found in the Designer menu. To add such an element to a template you can do the following:

1. Add an input element from the menu, for example a Text or Button.
2. Select the element in the template.
3. On the **Attributes** pane, select the desired input type from the **Type** drop-down list.

Changing a form element

Once an element has been added to a Form, it can easily be changed: simply select the element in the template, go to the the **Attributes** pane, and edit the element. An input element can even be changed to another type of input element by selecting the desired input type from the **Type** drop-down list.

Specifying a default value

Attribute a default value to a Text, Textarea and other Form elements by dragging a field from the Data Model pane directly onto the field, once it has been created. This also works when dragging a field from a detail table in a record set into a Form element that is contained within a Dynamic Table.

Note that the default value doesn't disappear when the user clicks the field, as placeholders do. To insert a placeholder in a field, type a label and choose **Use label as placeholder** as its style when adding the element to the form; see "Adding elements to a Form" on page 398.

Making elements required

To change the validation of a COTG or Form element, right-click the element and choose **Validation settings**. Now you can change the Form's validation method and set the requirements per field; see "Changing a Form's validation method" on page 531.

Grouping data using arrays

A Job Data File is an XML file created by a Workflow process upon submitting a Web Form or COTG Form. Grouping data in a Job Data File greatly simplifies both the Data Mapping workflow and looping over data in Designer scripts. A simple method to create arrays in that data file is to use **two pairs of square brackets** in the name of the form inputs. Put the name of the array between the first pair of square brackets. Between the second pair of square brackets, define the key to which the value belongs. Consider the following HTML form inputs:

```



```

The above HTML results in the following XML:

```

<values count="4">
  <user_account>pparker@eu.objectiflune.com</user_account>
  <name>Peter Parker</name>
  <company>Objectif Lune</company>
  <pinElm1>
    <pin_0>
      <left>122</left>
      <top>253</top>
      <type>dent</type>
    </pin_0>
    <pin_1>
      <left>361</left>
      <top>341</top>
      <type>dent</type>
    </pin_1>
  </pinElm1>
</values>

```

Note

To enable submitting arrays, you need to check the **Use PHP arrays** option in the HTTP Server user preferences in Workflow; see [Workflow Online Help](#).

In case multiple fields with the same name are encountered the previous value is overwritten. This way only a single occurrence of that field name will be available in the data containing the value of the last encountered occurrence of that field. This behaviour is also seen in the PHP language.

You can try out this feature with the COTG Time Sheet template, as explained in this how-to: [Using The PHP Array Option](#). The COTG Fields Table element (see "Fields Table" on page 524) in that template has an Add button to add rows to a table, and groups data following this approach.

Getting the status of unchecked checkboxes and radio buttons

Unchecked checkboxes and radio buttons are not submitted (as per standard HTML behavior), so how to get the state of those checkboxes and radio buttons? A common approach to get the state of unchecked checkboxes and radio buttons is to add a hidden field to the Form with the same name as the checkbox or radio button, for example:

```
<input type="hidden" name="status_1" value="0" />
<input type="checkbox" id="status_1" name="status_1" value="1" />
```

When multiple fields with the same name are encountered, the previous value is overwritten. This way the values for unchecked checkboxes and radio buttons can be processed easily.

Tip

The Capture OnTheGo (COTG) plugin automatically adds a hidden field for every unchecked checkbox on a Form when the Form is submitted. It does this for every Form; the template doesn't have to be a COTG template. (See: "Using the COTG plugin: cotg-2.0.0.js" on page 442.)

Using JavaScript

JavaScript files, libraries and frameworks can be added to a template, primarily for use in Web pages and Capture OnTheGo Forms.

Which kind of library or framework you'll want to work with depends on the type of features you really desire. For a bit of help with that and a few examples, see this how-to: [Using external libraries](#).

Some JavaScript files are added automatically:

- When you create a template with a COTG Template Wizard (see "Capture OnTheGo template wizards" on page 416), the Designer automatically adds the **jQuery** library and the COTG library: **cotg-2.0.0.js**. This also happens when you add a Capture OnTheGo (COTG) element to a template that you didn't start with a COTG template wizard. For more

information about this plugin, see "Using the COTG plugin: cotg-2.0.0.js" on page 442.

- Capture OnTheGo and Jumbotron template wizards automatically add the **Foundation** files v. 5.5.1 to the resources of the template. For more information about the use of Foundation in the Designer, see "Using Foundation" on page 420.

Tip

It is possible to open and edit any JavaScript file in the Designer: select **File > Open**, select **All files (*.*)** as the file type and then select a JavaScript file.

Adding JavaScript files to the resources

To add a JavaScript file to the resources:

- Right-click the **Javascript** folder on the **Resources** pane, and click **New Javascript**. Double-click it to open and edit it.
- Alternatively, drag and drop the JavaScript file from the Windows Explorer to the JavaScript folder on the Resources pane.

Next, include it in a Web page; see below.

Adding a remote JavaScript file

A Remote Javascript Resource is a file that is not located within your template but is hosted on an external web server (generally called a **CDN**). When generating Web output, these files are referenced in the web page's header and are served by the remote server, not by the Connect Server module.

There are a few advantages to using remote resources:

- These resources are not served by your server, saving on space, bandwidth and processing.
- Using a popular CDN takes advantage of caching - a client having visited another website using that same CDN will have the file in cache and not re-download it, making for faster load times for the client.

To add a remote javascript:

1. Right-click the **Javascript** folder on the **Resources** pane, and click **New Remote Javascript**.
2. Enter a name for the file as it appears in the Javascript resources. For better management, it's best to use the same filename as the remote resource.
3. Enter the URL for the remote resource. This must be a full URL, including the http:// or https:// prefix, domain name, path and filename.
4. Optionally, check **defer** or **async** to add the async or defer attribute to the <link> element in the <head> of the segment.
Defer postpones the execution of the script until the page has finished parsing. This attribute is required by APIs like Google Maps.
When **async** is checked, the script executes asynchronously with the rest of the page (while the page continues the parsing).
When neither option is checked, the script is fetched and executed immediately, while the parsing of the page is paused.
6. Optionally, for a Capture OnTheGo Form, you can check **Use cached Capture OnTheGo resource**, to prevent downloading a remote JavaScript file again if it has been downloaded before. The file should be available on a publicly accessible location, for example: a folder location on a corporate website, hosted by a CDN (Content Delivery Network) or shared via a Workflow process.

Note

In Workflow, when using the Create Web Content task, check the **Embed All Resources** option to download and embed all remote resources. (See Workflow Help: [Create Web Content](#).)

Popular hosted frameworks on CDN networks are:

- [jQuery on MaxCDN](#)
- [Zurb Foundation on CDNJS](#)
- [Bootstrap on MaxCDN](#)
- [Multiple frameworks on Google Developers](#)

Including a JavaScript file in a Web context

To link a JavaScript file to the Web context, or to a certain Web page template or COTG template:

1. On the **Resources** pane, expand the **Contexts** folder, and then either right-click the Web context, or expand the Web context and right-click a Web section.
2. Click **Includes**.
3. From the **File types** dropdown, select **JavaScripts**.
4. The available JavaScript files are listed at the left. Use the arrow buttons to move the JavaScript files that should be included to the right-hand list. Using the **Up** and **Down** buttons you can change the order of the files, too.
5. Click **OK**.

Using JavaScript in other Contexts

Email clients do not support JavaScript. Therefore, Email contexts cannot include JavaScript resources.

When a JavaScript file is included in a Print section, the Designer itself acts as the browser. When generating Print output, it runs the JavaScript after generating the main page flow contents and the pagination. So, it is possible to change the Print output by a JavaScript; for example, to add a barcode that includes the page number to each document. A warning is appropriate, however: changing the DOM may change the page flow and doing so at this point may result in bad output and/or serious errors or a crash of the software.

Capture OnTheGo

With the Designer you can create Capture OnTheGo templates. COTG templates are used to generate forms for the Capture OnTheGo mobile application. For more information about the application refer to these websites: [Capture OnTheGo](#) and [Capture OnTheGo in the Resource Center](#).

COTG Forms

A Capture OnTheGo Form is actually just a Web Form that has a number of characteristic features:

- Its **action** always specifies a Workflow HTTP Server Input task, so that when the Form is submitted, the form data is sent to the Workflow server. (See: "Specifying an action" on page 408.)

- It may contain special **COTG Input elements**, like a Signature, Geolocation, or Camera element. These require the COTG JavaScript library to be added to the template. This happens automatically when the Form is created with a COTG Template Wizard.
- Thanks to the mobile app, it may be used **offline**. The app will submit the Form data when a connection to the internet is available. Just make sure, if the Form uses remotely stored style sheets or JavaScript files, that the option 'Use cached Capture OnTheGo resource' is enabled when adding the resources to the template. This prevents that the app tries to download a file again that has already been downloaded.
- It may be **reusable**. This depends on a setting in the Output to Capture OnTheGo plug-in (found on the Connectors tab) in Workflow (see the Workflow Help: [Output to CaptureOnTheGo](#)). A reusable COTG Form is not deleted from the app's form library when it is submitted, so it can be used again.

Creating a COTG Form

A Capture OnTheGo Form is actually just a Web Form, so you could add a Form element to a Web page in the Web context without the use of a Template Wizard. It is strongly recommended however, to start the COTG Template using one of the COTG Template Wizards. They all include the appropriate JavaScript files and style sheets to create user-friendly, responsive forms; see "Capture OnTheGo template wizards" on page 416.

Before starting to create a COTG Form, take some time to structure the design process and to get familiar with the principles of form design, as explained in the topic "Designing a COTG Template" on page 413.

Reusable forms

Capture OnTheGo forms can be single-use or reusable. This doesn't depend on the design (although, of course, this should be reflected in the design). What makes a form reusable is a setting in the Output to Capture OnTheGo plugin in Workflow; see [Output to CaptureOnTheGo](#). In the Capture OnTheGo app a reusable form is called a 'template'.

Forms for offline use

Capture OnTheGo forms can be used offline. This is the case even when the form relies on remotely stored source files like JavaScript files and style sheets, provided that the option **Use cached Capture OnTheGo resource** is checked when adding them to the form.

Specifying an action

The **action** of the Capture OnTheGo Form element should specify a Workflow HTTP Server Input task (see Workflow Help: [HTTP Server Input](#)) that receives and handles the submitted data. The action will look similar to this: **http://192.168.175.1:8080/actionname** (where **actionname** is the HTTP action of the HTTP Server Input task).

For information about specifying an **action** for a Form, see "Adding a Form" on page 528 or "Changing a Form's properties" on page 530.

Note

For testing purposes, it is possible to use another URL for the Form's action or not to specify an action at all; see "Testing a Capture OnTheGo Template" on page 436.

Filling a COTG template

Before inserting elements in a COTG Form, have the design ready; see "Designing a COTG Template" on page 413.

In a Capture OnTheGo form, you can use special Capture OnTheGo Form elements, such as a Signature and a Barcode Scanner element. For a description of all COTG elements, see: "COTG Elements" on page 519. To learn how to use them, see "Using COTG Elements" on page 431.

Foundation, the framework added by the COTG template wizards, comes with a series of features that can be very useful in COTG forms; see "Using Foundation" on page 420.

Naturally, Web Form elements can also be used on COTG Forms (see "Forms" on page 527 and "Form Elements" on page 532) as well as text, images and other elements (see "Content elements" on page 465).

Capture OnTheGo templates can be personalized just like any other type of template; see "Variable Data" on page 604 and "Personalizing Content" on page 592.

Tip

Use the **Outline** pane at the left to see which elements are present in the template and to

select an element.

Use the **Attributes** pane at the right to see the current element's ID, class and some other properties.

Use the **Styles** pane next to the Attributes pane to see which styles are applied to the currently selected element.

Tip

Click the **Edges** button on the toolbar to make borders of elements visible on the Design tab. The borders will not be visible on the Preview tab.

Using JavaScript

COTG plugin

Capture OnTheGo widgets do not function without the COTG plugin: **cotg-2.0.0.js**. This plugin also makes it possible to add COTG Elements dynamically, set defaults for COTG elements, react to events that occur when a user interacts with a COTG element, etc. For more information see: "Using the COTG plugin: cotg-2.0.0.js" on page 442.

Foundation

For COTG templates created with a COTG Template wizard, lots of features are already available through the Foundation framework; see "Using Foundation" on page 420.

The Foundation JavaScript files and style sheets are only added automatically when you start creating a COTG template with a template wizard; see "Capture OnTheGo template wizards" on page 416.

Other JavaScript files

You may add other JavaScript files, libraries and frameworks to a template, to enhance your Capture OnTheGo Forms; see "Using JavaScript" on page 403.

Testing the template

A Capture OnTheGo (COTG) template will be used to create a form that can be downloaded, filled out and submitted using the COTG app. Before starting to actually use the template, you

will want to make sure that it produces a form that looks good and functions as expected. How to preview the form, how to submit data and how to preview the submitted data is described in another topic: "Testing a Capture OnTheGo Template" on page 436.

Sending the template to the Workflow tool

After testing the template (see "Testing a Capture OnTheGo Template" on page 436) the template must be sent to the Workflow module. Templates sent to the Workflow module can be used in any process within it.

How to send the template and the corresponding Data Mapping Configuration to the Workflow tool is explained in another topic: "Sending files to Workflow" on page 309.

Next, you can start building a Workflow configuration that receives and handles the submitted data. The configuration should start with a HTTP Server Input task (see Workflow Help: [HTTP Server Input](#)) of which the HTTP action is the one specified in the COTG Form's action.

Using COTG data in a template

When a user submits a COTG Form, a Workflow configuration may store the information in a database and/or push it into other Workflow processes, for example to send a letter or an email receipt. To be able to use the submitted data in a template for that letter or email receipt, follow these steps:

1. Get sample data

Before you can create a template that uses COTG data that is submitted from a certain COTG Form, you have to get access to a sample of that data. There are two ways to do this:

- Using the option **Get Job Data File on Submit** in Connect Designer; see "Testing a Capture OnTheGo Template" on page 436. This way you don't have to create a Workflow configuration first. Once the Job Data File is received by the Connect server, a dialog appears asking where to store it.
- Using a Workflow configuration. When a user submits a Capture OnTheGo Form, the data are received by a Workflow HTTP Server Input task (see Workflow Help: [HTTP Server Input](#)) that receives and handles the submitted data. Even when no other tasks are present in that Workflow configuration, Workflow can output an XML file that contains the submitted data, in a location specified for the Send To Folder plugin in Workflow.

Note

When a COTG Form is submitted, by clicking or touching the Submit button, the `name` and `value` of form elements are submitted. If a Checkbox or Radio Button is not checked, its name and value are not sent when the form is submitted.

Fortunately, there is a workaround for this; see "Using COTG Elements" on page 431.

The Form's validation should ensure that the data that the user submits is valid (see "Changing a Form's validation method" on page 531 and "How to make COTG elements required" on page 433).

2. Create a Data Mapping Configuration

Use the resulting XML file to create a Data Mapping Configuration (see "Data mapping configurations" on page 101).

1. Choose **File > New > Data mapping Wizards > From XML file**.
2. Select the XML data file as its source and click **Next**.
3. Set the XML Elements option to **/request/values**. This will automatically add an extraction step for the submitted form fields.
4. Click **Finish**. The file is opened in the DataMapper and the form fields are automatically extracted including the data for the signature and camera object.
5. Save the Data Mapping Configuration.

3. Create a template

Create a Designer template and personalize it using the Data Mapping Configuration (see "Personalizing Content" on page 592). Strings, base64-encoded strings and SVG data, stored in data fields using the DataMapper can be added to the template just like any other variable data; see "Variable Data" on page 604. They will show up in the template **as they are**.

Strings and base64-encoded strings show up as strings.

SVG data will be interpreted and displayed as an image.

Note

The Signature data returned by the COTG app (as of Android 10.6.0, iOS 10.6.0,

and Windows 6.0) is formatted so that the signature will automatically be scaled to fit in the containing box in a template. With previous versions of the app, the format of returned signatures could vary.

Adding Camera data to the template

The Camera widget submits a base64-encoded string, which can be put in a data field using the DataMapper. When this data field is dragged into a template, the string will show up in the content, instead of the image.

To make the image appear in a template, the data has to be used as the URL of an image. The below procedure describes how to use Camera data as an image inside a <div> container. The benefit of this approach is that the image automatically scales to the size of the container.

1. Click the **Insert Inline Box** icon on the toolbar. The **Insert Inline Box** dialog appears.
2. Enter an ID for the box (anything will do, as long as it helps you identify the box) and click **OK**. The box is added to the text flow and can be resized if needed.
3. Switch to the **Source** tab and replace the content of the box:

```
<p>
```

```
    Div content goes here
```

```
</p>
```

by this text:

```
<img id="camera" src="" width="100%">
```

4. Switch back to the **Design** tab. You will see a small, empty rectangle inside at the top of the inline box.
5. Right-click the empty rectangle and choose **New Script...** in the contextual menu. The **Edit Script** dialog appears. The selector of the script is automatically set to the ID of the selected element (`#camera`).
Alternatively, you could add a new script on the Scripts pane and make sure that the Selector field is set to `#camera`.
6. Enter the following script code:

```
results.attr("src", record.fields.photo);
```

The name of the data field (in this case: photo) must be that of the Camera data in your data model.
This script updates the attribute "src" with the field containing the base64 image.
7. Click **OK** to save the script and toggle to the **Preview** mode to see the result. You should see your image. When you resize the inline box that surrounds the image, the image

should be resized as well.

If the inline box isn't visible, click the Show Edges  button in the toolbar.

Designing a COTG Template

Designing a Capture OnTheGo template is more than adding elements to a Web form. This topic shares some insights regarding the design process and principles.

Design process

Ideally, the design process consists of the following steps.

1. **Gathering information.** It is often tempting to skip this step, especially when a Capture OnTheGo form replaces a paper form, but the research that you do to find out what the company actually needs will prove to be well worth your time. Creating specifications up front prevents discussions, reduces rework and therefore saves time.
2. **Listing the input fields** that are needed, their type, and possible input constraints. Think of how the information should be visually grouped. To get an overview of all the elements and features that can be used in a Capture OnTheGo form, check out the following pages:
 - "COTG Elements" on page 519, about elements that were specially designed for COTG.
 - "Form Elements" on page 532, about elements that can be used on COTG forms and on any other Web form.
 - "Using Foundation" on page 420, about elements and features that come with the Foundation framework that is added automatically by COTG Template wizards.
 - After creating a Capture OnTheGo template using a wizard, you can find more ready-made elements in the Snippets folder on the Resources pane.
3. **Creating mockups.** A mockup or wire frame will help you to layout the form and allows your customer to provide feedback early in the project. This will save you a lot of time: typically it is easier to change the sketch than to rework the code. In addition, mockups provide a way to do usability testing before actually creating the form. Note that mobile devices come in various sizes. It is important to adapt the form design to these screen sizes. There are various free and commercial mockup applications (both online and offline), but a sketch on paper will do too. Check out the free mockup templates from www.interfacesketch.com. Their templates are designed to help you sketch your designs for different devices on paper. Sketching tools and related techniques can be found on Zurb's website: [Sharpies](#), [Shaders](#) and [Highlighters](#).

4. **Creating the form.** Create the form in accordance with web design principles; see "Form design" below.
5. **Testing the form.** Even if you did proper research and showed a mockup, customers or users will likely come up with new requirements once they've seen the initial live version. Be prepared and plan for this, too.

Form design

Paper forms and web forms are very different in nature. For example, paper forms have a fixed size: the size of the paper they are printed on. Web forms can be viewed on screens with different sizes, in portrait or landscape format. Paper forms are filled out with a pen, while web forms are filled out using one's fingers or a stylus. Good form design requires an understanding on how users enter information on a mobile device and how they expect the form to look and behave.

Tip

If the COTG Form replaces a paper form, it can be tempting to stick to the original layout for the sake of recognizability. Don't fall into that trap. In the end, the users - customers and employees - will be happier with a user-friendly form that adapts to different screen sizes and looks like it was designed for the web.

Most design guidelines for web forms apply to COTG forms as well. Two key concepts are responsive design and usability.

Responsive design

Responsive Design is "an approach to web design aimed at crafting sites to provide an optimal viewing and interaction experience - easy reading and navigation with a minimum of resizing, panning, and scrolling — across a wide range of devices". (Source: Wikipedia.).

With the COTG app for Android or iOS, COTG forms can be viewed on a wide variety of mobile devices, with different screen sizes. A responsive design will adapt to the size and orientation of the screen, to avoid navigation tasks like zooming in or out and scrolling horizontally. The layout may change to optimize the user experience on that device: information that is shown side by side on a larger tablet may be stacked when viewed on a smaller device.

It is complicated and time consuming to create a responsive design all by yourself. Therefore it is advisable to start creating a COTG form with a COTG Template Wizard (see "Capture OnTheGo template wizards" on page 416). All Web and COTG Template Wizards in Connect Designer make use of the Zurb Foundation front-end framework to make the templates

responsive (see "Using Foundation" on page 420 and <http://foundation.zurb.com/learn/about.html>).

Tip



In the Designer, you can test the responsiveness of a form using the Responsive Design button at the top right of the workspace.

Some browsers also let you test the responsiveness of a form. In Firefox, for example, select Developer > Responsive Design to view a form in different sizes.

Usability

Usability defines the ease of use of a form. Is the layout intuitive? Are the form elements easy to tap on a mobile device? A visually consistent design allows the user to follow the flow while filling out the form. Below are some key usability aspects to keep in mind when designing forms.

Provide clear labels. Many modern web sites show labels inside the actual form inputs while they are empty. This saves space on the form, but once the user has entered data the label is no longer visible. Show a label at all times to help the user review his input.

Use font sizes that are big enough. On paper, smaller fonts are easier to read than on a web form. Of course, on a touch screen you can zoom in and out, but a user-friendly form doesn't force the user to do that.

Provide touch areas that are large enough. COTG forms are used on a mobile device (in the COTG app). Make sure that the user can easily tap the form elements, hyperlinks and buttons. The index finger of most adults covers an area that is between 45 and 55 pixels wide. There should be enough white space between the form inputs so the user won't accidentally put focus on the wrong element.

Visually group related information. Use headers to mark a section. This makes it easier to navigate the form. Applying a large font size and background color will make them stand out. You can use Foundation's off-canvas menu and accordion (collapse) functionality to make it easier to navigate groups of input fields.

Provide feedback. Show what input data is expected, clearly identify which fields are required and show errors when the entered data doesn't meet the required format.

Capture OnTheGo form characteristics

Reusable forms

Capture OnTheGo forms can be single-use or reusable. This doesn't depend on the design (although, of course, this should be reflected in the design). What makes a form reusable is a setting in the Output to Capture OnTheGo plugin in Workflow; see [Output to CaptureOnTheGo](#). In the Capture OnTheGo app a reusable form is called a 'template'.

Forms for offline use

Capture OnTheGo forms can be used offline. This is the case even when the form relies on remotely stored source files like JavaScript files and style sheets, provided that the option **Use cached Capture OnTheGo resource** is checked when adding them to the form.

Capture OnTheGo template wizards

With the Designer you can create Capture OnTheGo (COTG) templates. COTG templates are used to generate forms for the Capture OnTheGo mobile application. For more information about this application, see the website: [Capture OnTheGo](#).

A Capture OnTheGo Form is actually just a Web Form, that you could add without a wizard, but the COTG Template Wizards include the appropriate JavaScript files for the Capture OnTheGo app, and styles to create user-friendly, responsive forms. They are built upon the Foundation framework.

Foundation

All Web Template Wizards in Connect Designer make use of the Zurb **Foundation** front-end framework. A front-end framework is a collection of HTML, CSS, and JavaScript files to build upon. Foundation is a **responsive** framework: it uses CSS media queries and a mobile-first approach, so that websites built upon Foundation look good and function well on multiple devices including desktop and laptop computers, tablets, and mobile phones. Foundation is tested across many browsers and devices, and works back as far as IE9 and Android 2. See <http://foundation.zurb.com/learn/about.html>.

For more information about the use of Foundation in the Designer, see "Using Foundation" on page 420.

After creating a COTG template, the other contexts can be added, as well as other sections (see "Adding a context" on page 321 and "Adding a Web page" on page 388).

Tip

If the COTG Form replaces a paper form, it can be tempting to stick to the original layout. Although that may increase the recognizability, it is better to give priority to the user-friendliness of the form. Keep in mind that the COTG form will be used on a device and don't miss the chance to make it as user-friendly as possible. See "Designing a COTG Template" on page 413.

Creating a COTG template using a Wizard

To create a COTG template with a Template Wizard:

- In the **Welcome** screen that appears after startup and when you click the Home icon at the top right, choose **Browse Template Wizards**. Scroll down until you see the **Capture OnTheGo Starter** Template Wizards.
 - Alternatively, on the **File** menu, click **New**, expand the **Template** folder, and then expand the **Capture OnTheGo Starter** folder.
2. Select a template. There are 8 types of Web Template Wizards:
 - **Blank**. The Blank COTG Template has some basic design and the appropriate form, but no actual form or COTG elements.
 - **Bill of Lading**. The Bill of Lading Template is a transactional template that includes a detail table with a checkmark on each line, along with Signature and Date COTG elements. Use this wizard as a way to quickly start any new Zurb Foundation based form for Capture OnTheGo.
 - **Event Registration**. The Event Registration Template is a generic registration form asking for name, phone, email, etc.
 - **Event Feedback**. The Event Feedback Template is a questionnaire containing different questions used to rate an experience.
 - **Membership Application**. The Membership Application Template is a signed generic request form that can be used for memberships such as gyms, clubs, etc.
 - **Patient Intake**. The Patient Intake Template is a generic medical questionnaire that could potentially be used as a base for insurance or clinic form.

- **Kitchen Sink.** The Kitchen Sink Template includes a wide range of basic form and COTG form elements demonstrating various possibilities of the software.
 - **Time Sheet.** The Time Sheet Template is a single page application used to add time entries to a list. This template demonstrates the dynamic addition of lines within a COTG template, as the Add button creates a new time entry. There is no limit to the number of entries in a single page. Submitted data are grouped using arrays (see "Grouping data using arrays" on page 434).
3. Click **Next** and make adjustments to the initial settings.
 - **Create Off-Canvas navigation menu:** an Off-Canvas menu is a Foundation component that lets you navigate between level 4 headings (<h4>) in the form. Check this option to add the menu automatically.
 - **Submit URL:** enter the URL where the form data should be sent. The URL should be a server-side script that can accept COTG Form data.
 - The **Title** and the **Logo** that you choose will be displayed at the top of the Form.
 - **Colors:** Click the colored square to open the Color Picker dialog (see "Color Picker" on page 674) and pick a color, or enter a valid hexadecimal color code (see [w3school's color picker](#)) for the page background color. Do the same for the background color of the navigation bar at the top and for the buttons on the Form.
 4. Click **Next** to go to the next settings page if there is one.
 5. Click **Finish** to create the template.

The Wizard creates:

- A **Web context** with one web page template (also called a section) in it. The web page contains an 'off-canvas' Div element, Header, a Section and a Footer element with dummy text, and depending on the type of web page, a navigation bar, button and/or Form elements.
- **Style sheets** and **JavaScript files** related to the COTG form itself and others related to the Foundation framework (see above). The style sheets can be found in the Stylesheets folder on the Resources pane. The JavaScript files are located in the JavaScript folder on the Resources pane.
- A collection of **snippets** in the Snippets folder on the Resources pane. The snippets contain ready-to-use parts to build the web form. Double-click to open them. See "Snippets" on page 548 and "Loading a snippet via a script" on page 640 for information about using Snippets.

The Wizard opens the Web section, so that you can fill the Capture OnTheGo form.

6. Make sure to set the action and method of the form: select the form and then enter the action and method on the Attributes pane.

The **action** of a Capture OnTheGo form should specify the Workflow HTTP Server Input task that receives and handles the submitted data. The action will look like this:

http://127.0.0.1:8080/action (8080 is Workflow's default port number; 'action' should be replaced by the HTTP action of that particular HTTP Server Input task).

The **method** of a Capture OnTheGo form should be **POST** to ensure that it doesn't hit a data limit when submitting the form. The GET method adds the data to the URL, and the length of a URL is limited to 2048 characters. Especially forms containing one or more Camera inputs may produce a voluminous data stream that doesn't fit in the URL. GET also leaves data trails in log files, which raises privacy concerns. Therefore POST is the preferred method to use.

Filling a COTG template

Before inserting elements in a COTG Form, have the design ready; see "Designing a COTG Template" on page 413.

In a Capture OnTheGo form, you can use special Capture OnTheGo Form elements, such as a Signature and a Barcode Scanner element. For a description of all COTG elements, see: "COTG Elements" on page 519. To learn how to use them, see "Using COTG Elements" on page 431.

Foundation, the framework added by the COTG template wizards, comes with a series of features that can be very useful in COTG forms; see "Using Foundation" on the facing page.

Naturally, Web Form elements can also be used on COTG Forms (see "Forms" on page 527 and "Form Elements" on page 532) as well as text, images and other elements (see "Content elements" on page 465).

Capture OnTheGo templates can be personalized just like any other type of template; see "Variable Data" on page 604 and "Personalizing Content" on page 592.

Tip

Use the **Outline** pane at the left to see which elements are present in the template and to select an element.

Use the **Attributes** pane at the right to see the current element's ID, class and some other properties.

Use the **Styles** pane next to the Attributes pane to see which styles are applied to the currently selected element.

Tip

Click the **Edges** button on the toolbar to make borders of elements visible on the Design tab. The borders will not be visible on the Preview tab.

Using Foundation

This topic explains how to use the Foundation Grid and other Foundation components in a Web Form or COTG Form.

Foundation

All Web Template Wizards in Connect Designer make use of the Zurb **Foundation** front-end framework. A front-end framework is a collection of HTML, CSS, and JavaScript files to build upon. Foundation is a **responsive** framework: it uses CSS media queries and a mobile-first approach, so that websites built upon Foundation look good and function well on multiple devices including desktop and laptop computers, tablets, and mobile phones. Foundation is tested across many browsers and devices, and works back as far as IE9 and Android 2. See <http://foundation.zurb.com/learn/about.html>.

Capture OnTheGo and Jumbotron template wizards automatically add the Foundation files v. 5.5.1 to the resources of the template. In a future version of PlanetPress Connect, Foundation 6 will be included. If you'd rather start using the newest Foundation files right away, you have two options:

- Download the Foundation files (from <http://foundation.zurb.com/sites/download.html/>) and add them to the template manually.

- Use remote Foundation files from a CDN, such as <https://cdnjs.com/> (search for Foundation).

See "Using JavaScript" on page 403 and "Adding CSS files" on page 555 for further instructions.

Once the Foundation files have been added to a template, you can use the Grid, as well as many other Foundation components, in your template.

Tip

Take a look in the Snippets folder on the Resources pane. After creating a template with a Capture OnTheGo or Jumbotron template wizard, this folder contains a number of ready-made elements that make use of Foundation.

The Grid

Use the **Grid** to ensure the responsiveness of a form. Using the Grid essentially means building a form or web page with Div elements (a Div is a container element, see "Div" on page 516) that have the following `classes`:

- **row**: This class identifies a Div as a horizontal block (a row) that can contain up to 12 columns.
- **columns**: This class should be used for a Div inside a Div with the class `row`. It identifies a Div as part of a row Div.
- **small-n**, **medium-n**, **large-n**: These classes indicate the number of columns that this Div occupies within in the row, on a small, medium or large screen, respectively. Replace `n` with a number, for example: `small-2`, `large-4`. If the numbers declared in one 'row' for one screen size, added together, exceed the maximum of 12, they don't fit in one row on that screen size. In that case the Div elements will appear below each other instead of next to each other.

These classes can be combined, so that depending on the screen size, a Div can take more or less space in a row. Separate the class names with a space.

Tip

Start with the class for small screens. For example: `<div class="small-3 large-6" columns>`. Larger

devices will inherit those styles (thanks to the mobile-first approach of Foundation's style sheet). Customize for larger screens as necessary.

Example

This very simple layout has only one row:

```
<div class="row">
  <div class="small-2 large-4 columns">Content goes
here</div>
  <div class="small-4 large-4 columns">Content goes
here</div>
  <div class="small-6 large-4 columns">Content goes
here</div>
</div>
```

The Div elements inside the row take up 2, 4 and 6 parts of the total amount of screen size (divided in 12 parts) on a small screen. On a large screen they each take one third of the available space. If the class `large-4` would have been left out, the Divs would take up 2, 4 and 6 parts of the available space, regardless of the screen size.

There's more that you can do with the Grid, for example, you could center columns, or switch columns depending on the screen size they are viewed on. For information about all these possibilities, see this website:

<http://foundation.zurb.com/sites/docs/v/5.5.3/components/grid.html>.

Adding Divs and classes to a Connect Form template

To insert a Div, select **Insert > Structural Elements > Div** on the menu. To add a class to the Div, select the Div (see "Selecting an element" on page 469) and type the class in the Class field on the Attributes pane.

To add Grid rows and columns quickly, you could also use the **Grid** snippets or **Row** snippets, found in the **Snippets** folder on the **Resources** pane after using a wizard to create a Foundation web page or a Capture OnTheGo template. For more information about Snippets, see "Snippets" on page 548. For more information about template wizards, see "Creating a Web template with a Wizard" on page 382 and "Capture OnTheGo template wizards" on page 416.

Alternatively, If you are familiar with HTML, you can open the Source tab of the Workspace and simply type the HTML to add the Div elements and classes.

Tip

Use Emmet to create a Grid layout on the source tab really fast. See "Emmet" on page 362.

Other Foundation components

Foundation comes with many other components to improve and embellish Web forms and pages . A few examples:

- An **Accordion** can be used to expand and collapse content that is broken into logical sections, much like tabs. It can be very useful on long forms.
- An **Off-Canvas menu** lets the user navigate between level 4 headings (<h4>) in a Web page or form. Capture OnTheGo Template wizards offer the option to add this menu automatically.
- **Switches** are toggle elements that switch between an Off and On state on tap or click. They make use of checkbox inputs (or radio buttons) and require no javascript. Their size can be adapted, to make them easy to use on a touch screen.

For a full overview and explanation of all Foundation components (v. 5), see this web page: <http://foundation.zurb.com/sites/docs/v/5.5.3/>.

COTG Elements

With the Designer you can create Capture OnTheGo templates. COTG templates are used to generate forms for the Capture OnTheGo mobile application. This topic is about Capture OnTheGo form elements. For more information about the application refer to these websites: [Capture OnTheGo](#) and [Capture OnTheGo in the Resource Center](#).

Capture OnTheGo (COTG) elements can only be added within a Form element in a Web context; see "COTG Forms" on page 406. For information about how to add and use COTG Elements, see "Using COTG Elements" on page 431.

It is also possible to add COTG Elements dynamically, to set defaults for COTG elements and to react to custom events that occur when a user interacts with a COTG element. For more information see: "Using the COTG plugin: cotg-2.0.0.js" on page 442 and "Dynamically adding COTG widgets" on page 445.

Barcode Scanner

The Barcode Scanner element adds a button to trigger the device to scan a barcode. A very large variety of barcode types are supported (see the table below). Once the barcode has been scanned, the data from the barcode will be added to the COTG Form and submitted along with it.

Note

Using the Barcode Scanner element requires the installation of the [ZXing Barcode Scanner](#) application on Android devices. The application returns the barcode data after scanning.

Supported barcodes

Which barcodes are supported depends on the operating system of the device, (and, on iOS, possibly also on the actual version of the system).

Barcode Type	iOS	Android, Windows
Aztec Code	x	x
Codabar		x
Code 39	x	x
Code 93	x	x
Code 128	x	x
Data Matrix	x	x
EAN_8	x	x

Barcode Type	iOS	Android, Windows
EAN_13	x	x
I2OF5	x	
ITF	x	x
PDF417	x	x
QR CODE	x	x
RSS14		x
RSS_EXPANDED		x
UPC_A		x
UPC_E	x	x

Camera

The Camera element adds a group of buttons to capture or select an image. Once the image is selected via the camera or the device's library (aka "gallery"), it is saved within the Form data.

When the form is submitted, the image is sent in a base64-encoded string format. To learn how to add Camera data to a template, see "Adding Camera data to the template" on page 412.

The Camera element has a number of options, of which most can be set in the Design view. These options are described below.

All options, including options that cannot be set via the Design view, can be set via the Source view or in code; see "Changing default settings for widgets" on page 444.

Buttons

By default, the Camera element adds three buttons to the form:

- **Take now:** Opens the device's default Camera application to take a picture using the device's camera. Capture OnTheGo has no impact on the device's applications, so the features available (quality, orientation, filters) are dependent on the device itself. You can, however, set the format, quality and scaling for images that are submitted by the Camera element, as explained below.
- **Library:** Opens the device's default library or gallery application to select a single image that is then saved in the form data. The accessible images and navigation depend on the device itself.
- **Clear:** Removes any existing image data from the Camera element.

To omit the Take now or Library button, edit the Camera element's properties: right-click the Camera element after adding it to the form, select **Camera properties** and then use the **Source** drop-down to select which buttons will be available: Take, Pick from library, or both.

Annotations

Annotations can make a picture much more informative: an arrow, showing in which direction a car was driving; a circle, where the car has been damaged... To allow the user to make annotations, right-click the Camera element after adding it to the form, select **Camera properties**, and then check **Allow annotations**.

Clicking (or rather, touching) the image will bring up the annotation dialog. Annotations can be made in a Marker (semi-transparent) or Pencil (solid) style, in different colors and with different widths.

Annotations are submitted in SVG format by a hidden input added to the Camera element. The name of that input is the ID of the Camera element, followed by "-note-data", for example **camera1-note-data**.

Cropping/editing/deskewing

To allow the user to crop, edit and deskew the image after taking or selecting it, select **Camera properties**, and then check **Edit Image** and/or **Allow Deskew**. Which editing options the user actually gets and how they are presented to the user depends on the operating system of the device. On an Android device for example, the user may be able only to crop the image, while the user of an iOS device may select part of the image and rotate that selection.

Image format

You can set the format, quality and scaling for images that are submitted by the Camera element. Right-click the Camera element after adding it to the form, select **Camera properties**

and edit the Image properties:

- **Format:** The image format can be PNG or JPG.
- **Quality:** Set the compression in a percentage.
- **Scale Image:** Check this option to enable image scaling. Then set the maximum width and height of images before they are sent to the server. Note that only the smallest of these is applied and the size ratio is always maintained.

Time stamp

A time stamp can be added to each picture taken. Right-click the Camera element after adding it to the form, select **Camera properties**, and then check **Add Time Stamp**.

The time stamp will be added to the bottom left of the picture, with medium font size, and long date format (for example: 6/15/2009 1:45 PM). These settings can only be changed via the Source tab or in code; see "Using the COTG plugin: cotg-2.0.0.js" on page 442 and the Capture OnTheGo API: "Camera" on page 454.

Note

The Time stamp feature doesn't work in versions of the app prior to 10.6.

How to use the captured or selected image in a template

After a user has submitted the form and the data has been extracted, you may want to display the captured or selected image in a Designer template, for example in a letter or on a web page. To do this:

1. Load the data mapping configuration (or at least the data model).
2. Insert a dummy image in the template.
3. Right-click the dummy image and select **Dynamic Image**. The Text Script Wizard appears.
4. Under **Field** select the field that contains the base64-encoded string. The script puts the given string in the source (**src**) attribute of the image (****).

Instead of using the Text Script Wizard, you could also write a script yourself; see "Writing your own scripts" on page 624.

Date and Formatted Date

The Date element and the Formatted Date element display the current date on the device when the form is first opened. When the element is touched, a date selector appears so the user can modify this date. The Formatted Date element displays dates in a format that depends on the locale of the device on which the user is viewing the form. A Date Element displays dates in the ISO 8601 format: YYYY-MM-DD.

When the form is submitted, the date data is sent as plain text. A Formatted Date element submits the date in two formats: in the format that depends on the device's regional and language settings and in the ISO format mentioned above (using a hidden field). A Date element sends the date in the ISO format only.

Device Info

The Device Info Element adds a field that contains some information about the device (phone or tablet) that is submitting the COTG Form. This includes the device's type (Android or iOS), operating system version, device model and its UUID (Universally Unique Identifier). This information can be useful for both troubleshooting, if errors occur on specific device types for example, as well as for security validation: it is possible to maintain a list of device UUIDs that are allowed access, to prevent unauthorized use even if someone has a user name and password to a repository.

Document ID

The Document ID element retrieves the Document ID of the form currently viewed by the app. You could put the Document ID in a hidden input, so that when the form is submitted, the Document ID is submitted as well. A Document ID can be used on the server side to check (in the Connect database) if the data has already been submitted.

Fields Table

The Fields Table element adds a table with two rows, a delete button at the end of the first row and an add button at the end of the second row. Inside the rows you can put whatever elements you need. The user can click (or rather, touch) the Add button to add a row to the table. The new row will contain the same elements as the first row. The names of all elements in the first row will be extended with __0, while the names of the elements in the second row will be extended with __1, etc. (This means that the submitted data can be grouped; see "Grouping data using arrays" on page 434.)

Geolocation

The Geolocation Element adds a button to read the device's current GPS coordinates and save them in a form field. When the button is pressed, the GPS coordinates are requested and saved. When the form is submitted, the Geolocation data is sent in plain text.

High accuracy

By default, devices attempt to retrieve a position using network-based methods. To tell the framework to use more accurate methods, such as satellite positioning, the High Accuracy setting has to be enabled on the Geolocation element.

To make this setting, right-click the Geolocation element (or select it on the Outline pane) after adding it to the form, select **Geolocation properties** and check the **High Accuracy** option.

Note

The Geolocation element has several options, of which only one can be set via the user interface. All options, including those that cannot be set in Design view, can be set via the `data-params` attribute in the HTML, or in code; see "Using the COTG plugin: `cotg-2.0.0.js`" on page 442.

Image & Annotation

The Image & Annotation element is meant to be used with an image that needs input from the user. When inserting an Image & Annotation element you have to select the image. The user can simply click (or rather, touch) the image to bring up the annotation dialog. Annotations can be made in a Marker (semi-transparent) or Pencil (solid) style, in different colors and with different widths.

Annotations are submitted in SVG format by a hidden input. The name of that input is the ID of the Image & Annotation element, followed by "-note-data", for example **image_annotation1-note-data**.

Locale

The Locale Element does not have a UI element in the form. Inserting it adds a hidden input field that will contain the device's set locale when the form is submitted. This data is sent in plain text format and is available when processing the form data. The format is defined by the device.

Repository ID

The Repository ID element retrieves the repository ID (read only) from the app based on the currently logged on COTG user. You could put the Repository ID in a hidden input, so that when the form is submitted, the Repository ID is submitted as well. This information can be used on the server side to take specific actions, such as sending properly branded emails.

Signature

The Signature Element adds a signature box to a COTG form. These signatures are filled in via touch input, either with a finger or capacitive pen. Touching the signature box opens up a fullscreen box used to sign (generally more useful in Landscape mode depending on the device); after confirming, the dialog saves the data into the Form.

Signature data is transmitted in SVG plain text format. This type of data can be stored in a data field in a Data Mapping and dragged from the Data Model into a template as is. In Preview mode it will be displayed as an image because the Designer, just like web browsers, knows how to display this kind of data.

Note

The Signature data returned by the COTG app (as of Android 10.6.0, iOS 10.6.0, and Windows 6.0) is formatted so that the signature will automatically be scaled to fit in the containing box in a template. With previous versions of the app, the format of returned signatures could vary.

Time and Formatted Time

The Time element and the Formatted Time element display the current time on the device when the form is first opened. When the element is touched, a time selector appears so the user can modify this time. The Formatted Time element displays times in a format that depends on the locale of the device on which the user is viewing the form. A Time Element displays dates in the ISO 8601 format: HH:MM.

When the form is submitted, the time data is sent as plain text. A Formatted Time element submits the time in both the ISO format mentioned above and in the format that depends on the device's regional and language settings. A Time element sends the time in the ISO format only.

User Account

The User Account Element adds a hidden field that contains the Capture OnTheGo user account (an email address) that was used to submit the form to the server. This is useful if a

form is available to many different users, to detect who submitted it.

Using COTG Elements

Capture OnTheGo (COTG) elements are Web Form elements that are specially designed to be used in a Capture OnTheGo Form (see "Capture OnTheGo" on page 406). This topic explains how to add these elements to a Capture OnTheGo Form or and how to prepare them so that when the Form is submitted, they provide valid data that can be handled easily.

For a description of all COTG elements, see "COTG Elements" on page 519.

Adding COTG elements to a Form

To add a COTG element to a Form or Fieldset, click inside the Form or Fieldset, select **Insert > COTG elements**, and choose the respective element on the menu. Now you can change the element's settings:

1. Add an **ID** (required) and, optionally, a **class**.

Note

The ID will be copied to the `name` attribute of the element. The `name` attribute is what identifies the field to the receiving server-side script. To change the name, select the element after inserting it and type the new name on the **Attributes** pane.

ID's and classes are also useful with regard to variable data (see "Personalizing Content" on page 592) and styling (see "Styling templates with CSS files" on page 553).

2. Type a label, or choose No label under Style, to omit the label. (For Label elements there are no other options to be set.)
3. If applicable, choose a style for the label (for the label of a Checkbox, for example, you can't set a style).
 - **Wrap input with label** places the input element inside the Label element.
 - **Attach label to input** ties the label to the input element using the `for` attribute of the Label element.
 - **Use label as placeholder** inserts the given label text in the placeholder attribute of the field.

- **No style** omits the label altogether.

Note

The first two label styles ensure that when the user clicks the label, the input element gets the focus.

When you add a Capture OnTheGo (COTG) element to a template that you didn't start with a COTG template wizard, the Designer will automatically add the necessary JavaScript files: the jQuery library and the COTG library: `cotg-2.0.0.js`. (See: "Using the COTG plugin: `cotg-2.0.0.js`" on page 442.)

If a template contains an earlier version of the COTG library, the newest version will be added to the resources, but you will be asked which version of the library you prefer to use. Your preferred library will be included in the section (see: "Includes dialog" on page 682).

The Foundation JavaScript files and style sheets will not be added. You only get those automatically when you start creating a COTG template with a template wizard. (See: "Using Foundation" on page 420.)

Element specific settings

After inserting certain COTG elements, such as the Camera element, some important settings have to be made. These will appear when you **right-click** the element and select it from the short-cut menu.

Alternatively, you can make settings on the Source tab (see "Changing widget settings in HTML" on the next page).

The default settings can be changed in JavaScript; see "Changing default settings for widgets" on page 444.

Attributes

Some attributes (which aren't the same as the settings mentioned above) of a COTG element can be seen on the **Attributes** pane, after selecting the element (see "Selecting an element" on page 469).

All COTG elements have a `role` attribute. This attribute is not supposed to be edited: without the correct role attribute, the element won't function.

As noted, the `name` attribute is what identifies the element after submitting the form.

Tip

Use the **Outline** pane at the left to see which elements are present in the template and to select an element.

Use the **Attributes** pane at the right to see the current element's ID, class and some other properties.

Use the **Styles** pane next to the Attributes pane to see which styles are applied to the currently selected element.

Tip

Click the **Edges** button on the toolbar to make borders of elements visible on the Design tab. The borders will not be visible on the Preview tab.

How to make COTG elements required

To make a COTG element required, or to change the validation of a COTG Form, right-click the element and choose **Validation settings**. Set the Form's validation method to jQuery and set the requirements and a message per field. For an explanation see "Changing a Form's validation method" on page 531.

Changing widget settings in HTML

The Camera and Geolocation widgets have configurable options; you can decide which buttons will be visible in the Camera element, for example. A number of these options can be set via the user interface (see "Element specific settings" on the previous page).

The settings are stored in JSON format in the **data-params** attribute of the element in the HTML, as you can see on the Source tab in the Designer after making a setting.

All options, including options that cannot be set via the user interface, can be set via this `data-params` attribute. To define a timeout of 6 seconds for a Geolocation element, for example, you could switch to the Source tab and change its HTML to:

```
<div id="geolocation1" role="cotg.Geolocation" data-params="
```

```
{&quot;timeout&quot;:6000}">.
```

Settings in the HTML override the default settings for that element. They are applied to the widget when the Form is created and cannot be changed afterwards.

For a complete list of options see the Capture OnTheGo API: "Capture OnTheGo API" on page 453.

Settings for a **dynamically added** element can be made in code; see "Dynamically adding COTG widgets" on page 445.

The default settings that are specified in the COTG plugin can also be changed in code; see "Changing default settings for widgets" on page 444.

Grouping data using arrays

A Job Data File is an XML file created by a Workflow process upon submitting a Web Form or COTG Form. Grouping data in a Job Data File greatly simplifies both the Data Mapping workflow and looping over data in Designer scripts. A simple method to create arrays in that data file is to use **two pairs of square brackets** in the name of the form inputs. Put the name of the array between the first pair of square brackets. Between the second pair of square brackets, define the key to which the value belongs. Consider the following HTML form inputs:

```
<input type="hidden" name="user_account"
value="pparker@eu.objectiflune.com">
<input type="text" name="name" value="Peter Parker">
<input type="text" name="company" value="Objectif Lune">
<input type="text" name="pinElm1[pin_0][left]" value="122">
<input type="text" name="pinElm1[pin_0][top]" value="253">
<input type="text" name="pinElm1[pin_0][type]" value="dent">
<input type="text" name="pinElm1[pin_1][left]" value="361">
<input type="text" name="pinElm1[pin_1][top]" value="341">
<input type="text" name="pinElm1[pin_1][type]" value="dent">
```

The above HTML results in the following XML:

```
<values count="4">
  <user_account>pparker@eu.objectiflune.com</user_account>
  <name>Peter Parker</name>
  <company>Objectif Lune</company>
  <pinElm1>
    <pin_0>
```

```

        <left>122</left>
        <top>253</top>
        <type>dent</type>
    </pin_0>
    <pin_1>
        <left>361</left>
        <top>341</top>
        <type>dent</type>
    </pin_1>
</pinElm1>
</values>

```

Note

To enable submitting arrays, you need to check the **Use PHP arrays** option in the HTTP Server user preferences in Workflow; see [Workflow Online Help](#).

In case multiple fields with the same name are encountered the previous value is overwritten. This way only a single occurrence of that field name will be available in the data containing the value of the last encountered occurrence of that field. This behaviour is also seen in the PHP language.

You can try out this feature with the COTG Time Sheet template, as explained in this how-to: [Using The PHP Array Option](#). The COTG Fields Table element (see "Fields Table" on page 524) in that template has an Add button to add rows to a table, and groups data following this approach.

Getting the status of unchecked checkboxes and radio buttons

Unchecked checkboxes and radio buttons are not submitted (as per standard HTML behavior), so how to get the state of those checkboxes and radio buttons? A common approach to get the state of unchecked checkboxes and radio buttons is to add a hidden field to the Form with the same name as the checkbox or radio button, for example:

```

<input type="hidden" name="status_1" value="0" />
<input type="checkbox" id="status_1" name="status_1" value="1" />

```

When multiple fields with the same name are encountered, the previous value is overwritten. This way the values for unchecked checkboxes and radio buttons can be processed easily.

Tip

The Capture OnTheGo (COTG) plugin automatically adds a hidden field for every unchecked checkbox on a Form when the Form is submitted. It does this for every Form; the template doesn't have to be a COTG template. (See: "Using the COTG plugin: cotg-2.0.0.js" on page 442.)

Testing a Capture OnTheGo Template

A Capture OnTheGo (COTG) template will be used to create a form, that can be downloaded, filled out and submitted using the COTG app. Before starting to actually use the template, you will want to make sure that it produces a form that looks good and functions as expected. This topic explains how to preview the form, and how to submit data and preview the submitted data.

Previewing the form

On a PC

A Capture OnTheGo template can be previewed on a PC in two different ways. Note that Capture OnTheGo form elements will not be functional unless they are sent to a device.

- Within PlanetPress Connect Designer. You can open the **Preview** tab or the **Live** tab in the Workspace. This displays the output HTML along with any variable data being added. On the **Live** tab you can even fill out the form and submit it, and if the Get Job Data File on Submit option is enabled (via the toolbar of the same name), the Designer will receive an XML with the submitted data (see "Get Job Data File on Submit" on page 438). However, remember that COTG Form elements are only functional in the COTG app, so they won't submit any data.
- Within the default browser on your computer. Click the **Preview HTML** button in the toolbar. This opens your operating system's default browser and displays the form in that context.

Tip



In the Designer, you can test the responsiveness of a form using the Responsive Design button at the top right of the workspace.

Some browsers also let you test the responsiveness of a form. In Firefox, for example, select Developer > Responsive Design to view a form in different sizes.

Previewing a COTG Template in the app

A COTG Template cannot only be previewed on a PC; it can also be previewed on a mobile device. This will show the template within the Capture OnTheGo mobile application, and all widgets will be functional.

In order to test or use any Capture OnTheGo features you need to have a **Repository** account (also called a COTG Server account or the Store ID). You can get a trial account for this purpose; please see this page for more details: <http://www.captureonthego.com/en/promotion/>.

Once you have your Store ID and Password, you also need to create a **user** account:

1. Go to the Capture OnTheGo Repository Login: <https://config-us.captureonthego.com/>.
2. Login with your Store ID and Password.
3. Go to the Users page.
4. Add a new user. The user name should be in the form of an email address.

Next, make sure that the Capture OnTheGo mobile application is installed and that it is logged on as a known user of the Capture OnTheGo Repository.

Now, with your Capture OnTheGo template open in the Connect Designer module, click on the **Send COTG Test...** button in the toolbar.

Enter the appropriate information in the Send Test dialog (see "Send COTG Test" on page 722).

Click **Finish** to send the document. It should automatically appear in the app's Repository for 4 days from the moment it is sent. Once downloaded it remains accessible in the app's Library for 2 days.

Tip

To manually delete a test template from the app's Library, swipe it to the left.

Submitting and previewing data

When you hit the Submit button in a template in the **Designer** (on the Live tab), the submitted data can be sent back to the Designer in the form of an XML file (see below). The advantage of this is that you can immediately start creating a Data Mapping Configuration and use the data in a template.

Data submitted from the Capture OnTheGo **app** can be sent to you in the form of an email or saved via a Workflow configuration. Both options are explained below.

Note

The Form's validation should ensure that the submitted data is valid. Set the Form's validation method to jQuery and set the requirements and a message per field (see "Changing a Form's validation method" on page 531 and "How to make COTG elements required" on page 433).

Get Job Data File on Submit

It is possible to test a COTG Form **in the Designer** and get access to an **XML file** that contains the submitted data, without having a Workflow configuration to handle the data.

This option requires that:

- Workflow has been installed on the local machine, and the Workflow HTTP/Soap Service has been started. To do this, in the Workflow menu, click **Tools > Service Console**, then right-click **HTTP/Soap Server** and start it.
- In the Designer menu **Window > Preferences > Web**, the **Workflow URL** has been set to the correct host. The default is **http://127.0.0.1:8080/_getSampleFormData_**. This points to an internal process of the Workflow component running at that host.

If these conditions are met, you can get the XML file as follows:

1. Open the Form in the Designer, toggle to Live mode and fill out the form. Click the **Add Dummy Data** button (found on the toolbar and only available in the Live view) to populate empty form fields with dummy data. This adds dummy data to standard HTML form inputs as well as COTG inputs like the camera and signature widgets. Inputs that already contain data are left untouched. For a list of dummy data values, see the table below.

2. Click the **Get Job Data File on submit** toolbar button. This replaces the default form submit action and will send the form data to the Workflow's HTTP Service (which needs to be running in the background).

Note

Workflow's HTTP Service must be running, but not necessarily the Workflow Service itself.

3. Hit the **Submit** button. Now the data file will be sent directly to the Designer. Once the Job Data File is received by the Connect server, a dialog appears asking where to store it.
4. Save the XML file to disk. You can view it, create or update a Data Mapping Configuration for it (see "Data mapping configurations" on page 101), and insert the data in a template, using the Data Mapping Configuration (see "Personalizing Content" on page 592).

Note

Checkboxes and Radio buttons that are unchecked will not be submitted to the job data. This is standard behavior in HTML. One can work around that by adding a hidden field before the respective checkbox with the same name and for example value 0 (see "Using COTG Elements" on page 431).

Standard Form input dummy data values

Input	Dummy Value
Text	"Lorem ipsum dolor sit amet"
Textarea (multi-line text field)	"Lorem ipsum dolor sit amet"
Email	"pparker@localhost.com"
Number	random integer

Input	Dummy Value
Password	1234567890
URL	"http://www.localhost.com"
Checkbox	Checkboxes in detail tables and in the Fields Table control (time sheet) are checked.
Radio button	Selects the first radio button that is not disabled in each radio group. The radio group will be left untouched when there is a selected radio button.

Capture On The Go input dummy data values

Input	Dummy Value
Signature	Receives SVG signature data and the onscreen presentation of that data.
Camera	A dummy foto is added, and a (SVG) annotation if that option is set for the widget. Note that the script doesn't look at the PNG/JPG or resolution options, the only option it considers is the annotation option.
Geolocation widget	A geo location
Date Picker (formatted and standard)	Today's date*
TimePi	Now*

Input	Dummy Value
cker (format ted and standa rd)	
Device Info widget	" {\"available\":true,\"platform\":\"Android\",\"version\":\"9.9.9\",\"uuid\":\"17206724b8077491\",\"cordova\":\"3.6.4\",\"model\":\"Connect Designer\"}"
User Accou nt widget	"user@localhost.com"
Locale widget	en-US

* Note that the **formatted** date and time can be different from the values that the COTG app provides. In the COTG app the formatted date comes from the COTG API, and the formatted date and time normally depend on the locale/region settings on the mobile device. The **ISO** date and time should be the same as when using the COTG app.

Get submitted data via email

Getting submitted data via email requires the Form's action to be set to a test URL that contains an **API Key**. You can obtain an API Key as follows.

1. Go to <http://learn.objectiflune.com/>.
2. Create an account, or log in to your account.
3. Go to your Profile Page, and click the API Key link.

Now, when creating or editing a COTG Form, you can use the API Key in the Form's action:

1. Select the Form (see "Selecting an element" on page 469).
2. On the **Attributes** pane, paste the following URL in the **action** field:
http://learn.objectiflune.com/services/cotg-debug?__ol__auth_key={{APIKEY}}.
3. Replace {{APIKEY}} by your API Key.

When you submit the form in the COTG app (see "Previewing a COTG Template in the app" on page 437), the debug service will compose an HTML email that contains the form element names and the submitted values. Image files, like pictures and signatures, are added to the email as attachments. The email will be sent to the email address that you provided via your Learn profile.

For a more detailed description of this test procedure, see this how-to: [Testing a COTG template](#).

Get submitted data via Workflow

Eventually, when a user submits a Capture OnTheGo Form, the data are received by the Workflow HTTP Server Input task (see Workflow Help: [HTTP Server Input](#)) that has the same HTTP action as the one specified in the Form's action (see "COTG Forms" on page 406). The Workflow configuration should then handle the submitted data. But even if it doesn't, when no other tasks are present in that Workflow configuration, Workflow can output an XML file that contains the submitted data and save it in a location specified for the **Send To Folder** plugin in Workflow.

This XML file contains the actual data submitted by all Form elements, including COTG elements, such as a signature or barcode.

Using the COTG plugin: cotg-2.0.0.js

A Capture OnTheGo (COTG) Form may contain special COTG input elements, like a Signature, Geolocation, or Camera element. These elements do not function without the **COTG JavaScript library**. It is this library that links the controls with hardware features on the mobile device.

As of Connect 1.8, **cotg-2.0.0.js** replaces the cotg-1.x.js versions of the library. The new COTG plugin introduces **options** and **custom events** for COTG widgets. This greatly simplifies event-based programming in Capture OnTheGo Forms. For example: your code can now automatically set a date for a Date field and retrieve the Geolocation as soon as a form has been signed.

All available options and events are listed in the Capture OnTheGo API: "Capture OnTheGo API" on page 453.

This topic explains in detail how to add the COTG plugin, how to change the defaults for COTG widgets and how to use events.

How to add COTG elements to a Form dynamically is explained in another topic: "Dynamically adding COTG widgets" on page 445.

It is assumed that you have a basic understanding of HTML forms, CSS, JavaScript, and jQuery. Examples on this page use jQuery.

About jQuery

This version of the COTG library is entirely based on jQuery. jQuery is a JavaScript library that makes it very easy to select elements in a web page using HTML and CSS selectors, and to manipulate those elements. You will need to use jQuery to dynamically add widgets to a COTG Form. If you are new to it, spend a few minutes on learning it - it's that easy. For more information, see: <https://jquery.com/>. and <http://learn.jquery.com/>.

Adding the plugin

When you create a template with a COTG Template Wizard (see "Capture OnTheGo template wizards" on page 416), the Designer automatically adds the jQuery library and the COTG library: `cotg-2.0.0.js`.

This also happens when you add a Capture OnTheGo (COTG) element to a template that you didn't start with a COTG template wizard.

If a template contains an earlier version of the COTG library, the newest version will be added to the resources, but you will be asked which version of the library you prefer to use. Your preferred library will be included in the section (see: "Includes dialog" on page 682).

When this library is included in a Web template instead of a COTG template, it won't affect the template, except when the user submits a Form. At that moment the plugin will automatically add a hidden field for every unchecked checkbox on the Form.

Tip

If you want to take a look at the contents, you can open the plugin within the Designer: double-click **`cotg-2.0.0.js`** on the Resources pane.

Changing default settings for widgets

The Camera and Geolocation widgets have options that you can configure per element. You can decide, for example, which buttons will be visible in a specific Camera element (see "Element specific settings" on page 432).

The **default** settings for these options are specified in the COTG plugin. It is possible to change these defaults without modifying the plugin itself.

To do that, create a JavaScript file (see "Adding JavaScript files to the resources" on page 404) and specify the desired default settings in that file like this:

```
$.fn.widget.defaults.setting = value;
```

Make sure to include your JavaScript file in the Web context or in the Web section that contains the COTG Form (see "Including a JavaScript file in a Web context" on page 405).

All available settings are listed in the Capture OnTheGo API: "Capture OnTheGo API" on page 453.

Example

The following code sets the default timeout and accuracy for Geolocation objects, and the default maximum height and width for Camera widgets.

```
$.fn.cotgGeolocation.defaults.timeout = 6000; // 6 secs
$.fn.cotgGeolocation.defaults.enableHighAccuracy = true;
$.fn.cotgPhotoWidget.defaults.width = 1024;
$.fn.cotgPhotoWidget.defaults.height = 768;
$.fn.cotgPhotoWidget.defaults.quality = 60;
```

Reacting to, or triggering, widget events

The new COTG plugin introduces custom events for COTG controls. You can trigger and/or react to them as the user interacts with the Form.

- Use jQuery's **.on()** method to attach an event handler to an element (or set of elements). Call this function on the `$(document).ready` event, which is triggered when the Form is loaded in the app.
- Use the **.trigger()** method to trigger an element's event.

The events of all COTG widgets are listed in the Capture OnTheGo API: "Capture OnTheGo API" on page 453.

Examples

The sample below attaches an event handler to the "set" event of a Signature element. Once the signature is set (that is, after the user has clicked the Done button), the event handler triggers events of the Date and Geolocation elements. The Date element is set with today's date and the Geolocation element is updated with the current location.

```
$(document).ready(function() {
    $('#signature').on("set.cotg", function() {
        $("#date").trigger("set.cotg", new Date()); // set current
        $("#geolocation").trigger("update.cotg"); // get current
    });
});
```

The following example invokes the Date dialog when the user clicks a button.

```
$(document).ready(function() {
    $('#my-datepicker-button').on('click', function() {
        $('#date1').trigger("show-date-picker.cotg", new Date("2018
01"));
    });
});
```

Dynamically adding COTG widgets

Capture OnTheGo (COTG) widgets can be added to a Form dynamically, via jQuery. For example: a new Camera element could be added when the user clicks an Add button. This topic explains how to implement this. It is assumed that you have a basic understanding of HTML forms, CSS, JavaScript, and jQuery.

Prerequisites

Before you can start writing code that adds a widget in response to an action of the user, you need the following:

- Some element on the Form to trigger the creation of the widget. This could be anything that responds to an action of the user; a button or link, for example. Make sure that this element has an ID.
- A Form element to which the new widget can be added; a <div> for example. Again, make sure to give this element an ID.

- The HTML structure and attributes of the widget, so that you can recreate it in code. The HTML structure of a widget can be seen on the Source tab after inserting the same kind of widget into a Form in the Designer.

Also, if you don't have a JavaScript file for your code yet, add one to the resources of your template (see "Adding JavaScript files to the resources" on page 404) and make sure to include that file in the Web context or in the Web section that contains the COTG Form (see "Including a JavaScript file in a Web context" on page 405).

Adding an event handler

First of all you need to write an event handler that responds to the event that is meant to trigger the creation of the widget (e.g. the `onClick` event of a button or link), by calling the function that creates the widget. The event handler has to be added on the `$(document).ready()` function, which is fired when the Form is loaded in the browser/app.

```
$(document).ready(function() {  
    $('#add-element').on('click', function() {  
        //call the function that creates the widget  
    });  
});
```

Creating the widget

Now you can start writing the code that constructs, adds and initializes the widget. This code has to be based on jQuery.

Constructing the HTML

A widget basically is an HTML element with certain attributes and contents. The HTML structure of a widget can be seen on the Source tab after adding the widget to a Form in the Designer. In code, reconstruct the HTML. Make sure to give the new element an ID.

This code constructs the HTML of a Date element:

```
<label>date1  
    <input id="date1" role="cotg.DatePicker" readonly="" name="date1"  
    type="text">  
</label>
```

Adding it to the Form

Add the HTML to an element on the Form using the `append()` function.

The following code appends the contents of the variable `html` to an element on the Form that has the ID `cameras`:

```
$('#cameras').append(html);
```

Initializing the widget

A widget has to be initialized to allow it to be actually used. Widgets that are already present in a Form at startup are initialized as soon as the Form is loaded in the app. A widget that has been added to a Form dynamically has to be initialized directly after adding it to the Form; otherwise the new widget won't respond to actions of the user, even though it is visible in the app.

Initializing a widget takes just one line of code, in which you select the new widget by its ID and call the initialization function on it. This code, for example, initializes a new Camera element that has the ID `myCamera`:

```
$('#myCamera').cotgPhotoWidget();
```

Optionally, while initializing an element, you can make settings for this specific element. These settings get prevalence over the options already specified in the HTML and over the default settings specified in the COTG plugin.

The code snippet below initializes a new Camera element (with the ID `myCamera`) with a number of settings:

```
$('#myCamera').cotgPhotoWidget({
    quality: 50,
    height: 1024,
    width: 1024
});
```

The initialization functions and options of all widgets are listed in the Capture OnTheGo API: "Capture OnTheGo API" on page 453.

To learn how to set the **defaults** for one kind of elements, see "Changing default settings for widgets" on page 444.

Restoring a widget

When a Form is closed, the app stores the values of input fields to the local repository of the app, but the values of input fields of dynamically added widgets are not stored.

When you reopen the Form the original input fields and their values are restored. However,

dynamically added widgets are not restored; this needs to be handled in code. How to do this is explained in another topic: "Saving and restoring custom data and widgets" on the next page.

Example: adding Camera widgets dynamically

The following code inserts a Camera widget when the user clicks on a link or button with the ID `add-camera`. The `addCameraWidget()` function creates and adds the widget. Each new widget gets the class `camera-dyn`. The number of input elements that have this class is used to construct a unique ID for each new Camera widget.

The HTML structure of the widget was copied from the Source tab after inserting a Camera widget to a Form in the user interface of the Designer. The `addCameraWidget()` function appends this HTML to a `<div>` with the ID `cameras`, which was already present in the form. Subsequently the widget is initialized so that it is linked to the COTG app and the hardware features of the device.

```
$(document).ready(function() {
    $('#add-camera').on('click', function() {
        var cameraID = "camera_" + getCameraIndex();
        addCameraWidget(cameraID);
    });
});

function getCameraIndex(){
    return $("input.camera-dyn").length;
}

function addCameraWidget(cameraID, value) {
    if(typeof value == 'undefined') {
        value = '';
    }
    var html = '<label>Camera' +
        '<div id="' + cameraID + '" role="cotg.PhotoWidget">' +
        '<div class="panel" role="control-wrapper"
style="position:relative;">' +
        '<img role="photo" src="">' +
        '<input role="photo-data" class="camera-dyn" name="' + cameraID +
        '" type="hidden" value="' + value + '">' +
        '</div>' +
        '<button class="small" role="take-button" type="button">Take
now</button>' +
        '<button class="small" role="pick-button"
type="button">Library</button>' +
        '<button class="small" role="clear-button'
```

```

type="button">Clear</button>' +
    '</div></label>';
$('#cameras').append(html); // add the camera object to the DOM
$('#' + cameraID).cotgPhotoWidget(); // init COTG widget
}

```

Saving and restoring custom data and widgets

The Capture OnTheGo (COTG) app stores the values of the input fields to the local repository of the app upon closing the Form. On reopening the Form the original input fields and their values are restored. However, dynamically added widgets (see "Dynamically adding COTG widgets" on page 445) don't get restored. This needs to be handled in code. This topic explains how to do that.

Adding event listeners

In the process of closing and opening a Form two important events are triggered by the COTG library:

- **olcotgsavestate**. This event is meant for custom scripts to save information when the user closes or submits the Form. It occurs after the state of all static input fields has been saved.
- **olcotgrestorestate**. This event allows custom scripts to do something when the Form is reopened. It occurs after all static inputs have been restored.

To latch on to these events, you have to register a `olcotgsavestate` listener and a `olcotgrestorestate` listener, respectively, on the `window` object. The `window` object represents a window containing a DOM (document object model), in this case the COTG Form.

The function callback on its `addEventListener` method allows you to implement custom code as part of the save and restore processes.

(For more information about the `addEventListener` method see the documentation: [Mozilla](#) and [w3schools](#).)

The following code registers the `olcotgsavestate` listener on the `window` object.

```

window.addEventListener('olcotgsavestate', function(event) {
/* code to save custom data */
});

```

The following snippet does the same for the restore process.

```
window.addEventListener('olcotgrestorestate', function(event) {  
  /* code to retrieve custom data and restore custom widgets */  
});
```

The following code combines the previously saved string with the restored URL of a (static) COTG Camera element:

```
window.addEventListener("olcotgrestorestate", function(event) {  
  var value = event.detail.state["myString"];  
  value = value + $("#camera1 img").attr("src");  
  $("#form p").html(value);  
});
```

Note

You should not register for these events after the document has loaded (e.g. on the `$(document).ready()` event), because these events might get triggered before the document is ready.

Place your code in a separate JavaScript file and make sure to include that file in the Web context or in the Web section that contains the COTG Form (see "Including a JavaScript file in a Web context" on page 405). For this file, the order in which JavaScript files are included in the template doesn't matter.

Handling the save and restore events

The code inside the function callback of the added event listeners should respond to the event, by saving or retrieving custom information, respectively.

When the COTG app saves the Form, you can store extra information in the `event.detail.state` object as follows:

```
event.detail.state["key"] = value;
```

The **key** can be any string, as long as that string isn't the same as the ID of one of the widgets on the Form.

In the code that handles the `olcotgrestorestate` event, use the same key to retrieve the stored information.

The following sample saves the value "test" using "myString" as the key, when the Form is saved:

```
window.addEventListener("olcotgsavestate", function(event) {
    event.detail.state["myString"] = "test";
});
```

The following code retrieves the value that was stored with the `myString` key and displays it in a paragraph (a `<p>` element) at the top of the form.

```
window.addEventListener("olcotgrestorestate", function(event) {
    var value = event.detail.state["myString"];
    $("form p").html(value);
});
```

Note

When you've used jQuery to register for the events - `$(window).on()` - you must use `event.originalEvent` in the event handler functions, for example:

```
$(window).on("olcotgsavestate", function(event) {
    event.originalEvent.detail.state["mywidget"] = "test";
});
```

Restoring widgets

When a COTG Form is reopened, the app restores all input fields and widgets that were already present in the original Form (i.e. the Form deployed via Workflow or sent as test using the Designer). Dynamically added widgets don't get restored. To restore dynamically added widgets, you have to:

- **Save** information that reveals which widgets were added dynamically, and save the values of their input fields.
This code should go in the event handler for the `olcotgsavestate` event.
- **Add and initialize** the widgets again after the Form is reopened. Make sure to put any saved values back in the HTML.
This code should go in the event handler for the `olcotgrestorestate` event.
- **Trigger** the `restore-state.cotg` event on the newly added widget, to make sure that it is displayed correctly. For example:

```
$('#myCamera').trigger('restore-state.cotg',  
event.detail.state);
```


Put this code in the event handler for the `olcotgrestorestate` event.

Note that the widget must have the same ID as before in order to be able to retrieve its state.

For a detailed example, see: "Saving and restoring Camera widgets" below.

Example

Saving and restoring Camera widgets

This example demonstrates a way to save and restore dynamically added Camera widgets. How to add Camera widgets is explained in another topic: "Dynamically adding COTG widgets" on page 445.

When a user takes or selects a picture with a Camera widget, the COTG app stores the path to the image that was taken or selected in a hidden input. This is the path to the image on the device that runs the COTG app. On submitting the Form the COTG app replaces this value - the path - with the actual image data.

In this example the hidden fields of dynamically added Camera elements have got the `.camera-dyn` class. In the event handler for the `olcotgsavestate` event, the jQuery `each()` method is used to iterate over all inputs that have this class, storing their name and value in an object. Next, this object is stored - in JSON format - in the `event.detail.state` data with the key `my-camera-data`.

```
window.addEventListener('olcotgsavestate', function(event) {
    var camObj = {};
    $('input.camera-dyn').each(function() {
        var camera = $(this).attr('name');
        var val = $(this).val();
        camObj[camera] = val;
    });
    event.detail.state['my-camera-data'] = JSON.stringify(camObj);
});
```

In the `olcotgrestorestate` event handler the previously stored JSON is read from the `event.detail.state` and parsed into a JavaScript object. A `for ... in` loop is then used to iterate over the keys (the camera names) in that object. Inside this loop the cameras are added by calling the `addCamera()` function with the ID and value (the path to the picture) of each Camera element. Subsequently the `restore-state.cotg` event of the new widget is called to make sure that the thumbnail of the picture is shown and the Clear button becomes visible.

```
window.addEventListener('olcotgrestorestate', function(event) {
    var json = JSON.parse(event.detail.state['my-camera-data']);
    for (var cameraID in json) {
```



```

        var value = json[cameraID];
        addCameraWidget(cameraID,value);
        $('#'+ cameraID).trigger('restore-state.cotg',
event.detail.state);
    }
});

function addCameraWidget(cameraID, value) {
    if(typeof value == 'undefined') {
        value = '';
    }
    var html = '<label>Camera' +
'<div id="" + cameraID + "" role="cotg.PhotoWidget">' +
'<div class="panel" role="control-wrapper"
style="position:relative;">' +
'<img role="photo" src="">' +
'<input role="photo-data" class="camera-dyn" name="" + cameraID +
"" type="hidden" value="" + value + "">' +
'</div>' +
'<button class="small" role="take-button" type="button">Take
now</button>' +
'<button class="small" role="pick-button"
type="button">Library</button>' +
'<button class="small" role="clear-button"
type="button">Clear</button>' +
'</div></label>';
    $('#cameras').append(html); // add the camera object to the DOM
    $('#'+ cameraID).cotgPhotoWidget(); // init COTG widget

```

Capture OnTheGo API

As of Connect 1.8, **cotg-2.0.0.js** has replaced the cotg-1.x.js versions of the Capture OnTheGo (COTG) plugin, introducing **events** and **options** for COTG widgets. This topic lists all available options and custom events for widgets, as well as their initialization function.

How to use the COTG plugin is explained in the following topic: "Using the COTG plugin: cotg-2.0.0.js" on page 442.

To learn how to create widgets in code, see "Dynamically adding COTG widgets" on page 445 and "Saving and restoring custom data and widgets" on page 449.

For a list of all COTG elements and their intended use, see "COTG Elements" on page 519.

Barcode Scanner

cotgBarcode()

Initializing a Barcode Scanner element prepares it for user interaction.

Example: `$('#myScanner').cotgBarcode();`

Events

The Barcode Scanner **listens for** the following events.

Event	Description
clear.cotg	Removes the scanned Barcode data.
scan.cotg	Opens the scanner.

The Barcode Scanner **broadcasts** the following events.

Event	Description
set.cotg	This event is fired after Barcode data has been set to the value of the input.

Camera

cotgPhotoWidget([options])

options

Optional. An array containing the desired settings, e.g. {quality: 50, height: 1024, width: 1024}. For any unspecified options the default settings will be used.

Initialize the new Camera element with any settings that you want to be different from the defaults.

Example: `$('#myCamera').cotgPhotoWidget({quality: 50, height: 1024});`

How to change the default settings is explained in another topic: "Changing default settings for widgets" on page 444.

Option	Description	Type	Default
editimage	Allows the user to edit the image after taking or selecting it. Which editing options the user actually gets and how they are presented depends on the operating system of the device.	Boolean	false
encodingtype	The returned image's file encoding: <code>jpg</code> or <code>png</code> .	String	'jpg'
height	The maximum height in pixels	Number	864
width	The maximum width in pixels.	Number	1152
source	Which buttons are enabled: Take now (<code>take</code>), Library (<code>pick</code>), or both (<code>takeandpick</code>).	String	'takeandpick'
scaleimage	Scales the image to fit the maximum width or height. The aspect ration is maintained.	Boolean	true
quality	Quality of the saved image, expressed as a range of 0-100, where 100 is full resolution with no loss from file compression.	Number	80
allowannotations	Adds an Image & Annotation widget to edit the picture.	Boolean	false
allowdeskew	Allows the user to perform a perspective cropping after taking or selecting a picture.	Boolean	false
addtimestamp	Adds a time stamp	Boolean	false
stampFormat	The date format. For all possible formats	String	'L'

Option	Description	Type	Default
	see Cordova documentation . 'L' stands for localized date and time.		
stampAlign	The position of the stamp, specified by combining horizontal alignment (left, center, right) with vertical alignment (top, middle, bottom) and joining them with a vertical bar (' ').	String	'bottom left'
stampFontSize	The time stamp's font size: <code>small</code> , <code>medium</code> , or <code>large</code> .	String	'medium'

Events

The Camera **listens for** the following custom events.

Event	Description
clear.cotg	Removes the picture.
save-state.cotg	Saves the path of the current picture to the local storage of the COTG app.
restore-state.cotg	Restores the state of the widget when the form is reopened in the COTG app, after the app has restored previously entered values of static input fields.

The Camera **broadcasts** the following events.

Event	Description
clear.cotg	Fired after the user has clicked the Clear button.
set.cotg	Fired after the user has taken or selected a picture.

Date and Formatted Date

cotgDatePicker()

Initializing a Date or Formatted Date element prepares it for user interaction.

Example: `$('#myDatePicker').cotgDatePicker();`

Note that the difference between a Date and a Formatted Date is laid down in the HTML structure of the element.

Events

The Date and Formatted Date elements **listen** for the following custom events.

Event	Description
clear.cotg	Removes the date.
set.cotg	Sets the given date. The date should be given as a Date object, for example: <code>\$('#date').trigger("set.cotg", new Date()); // set current date</code>
show-date-picker.cotg	Opens the Date picker. Optionally, you can provide a date (as a Date object) for the Date picker to be opened with, for example: <code>\$('#date1').trigger("show-date-picker.cotg", new Date("2018-01-01"));</code>

Device Info

cotgDeviceInfo()

Initializing a Device Info element puts information about the device (phone or tablet) that displays the form, in the hidden input of the element.

Example: `$('#myDeviceInfo').cotgDeviceInfo();`

Events

The Device Info element **listens for** the following event.

Event	Description
clear.cotg	Removes the Device Info data.

The Device Info element **broadcasts** the following event.

Event	Description
set.cotg	This event is fired during initialization of the element, after setting its info to the current device.

Document ID

cotgDocumentId()

Initializing a new Document ID puts the current Document ID in the hidden input of the element.

Example: `$('#myDocID').cotgDocumentId();`

Events

The Document ID element **listens for** the following event.

Event	Description
clear.cotg	Removes the Document ID.

The Document ID element **broadcasts** the following event.

Event	Description
set.cotg	This event is fired during initialization of the element, after setting its value to the current Document ID.

Fields Table

The Fields Table element itself is just a table, so it doesn't have to be initialized. However, after dynamically adding it to a Form, you still have to add the correct event handlers to the Add and Delete buttons, as follows (replace `myTable` by the ID of your table):

```
$("#myTable [role=cotg-add-row-button]").on("click", function(){
    $(this).closest('table').cotgAddRow(true);
});

$("#myTable").on("click", "[role=cotg-delete-row-button]", function
(){
    $(this).closest('tr').cotgDeleteRow();
});
```

Geolocation

`cotgGeolocation([options])`

options

Optional. An array containing the desired settings, e.g. {enableHighAccuracy: true, timeout: 3000}. For any unspecified options the default settings will be used.

Call `cotgGeolocation([options])` on the new Geolocation element with any settings that you want to be different from the defaults.

Example: `$('#myGeolocation').cotgGeolocation({timeout: 6000});`

How to change the default settings is explained in another topic: "Changing default settings for widgets" on page 444.

Option	Description	Type	Default
enableHighAccuracy	By default, the device attempts to retrieve a position using network-based methods. Setting this property to true tells the framework to use more accurate methods, such as satellite positioning.	Boolean	false
maximumAge	Accept a cached position if it isn't older than the specified time in milliseconds.	Number	3000

Option	Description	Type	Default
timeout	The maximum length of time (milliseconds) that is allowed to pass before the position is retrieved.	Number	2700

Events

The Geolocation element **listens for** the following events.

Event	Description
clear.cotg	Removes the Geolocation data.
restore-state.cotg	Restores the state of the widget when the form is reopened in the COTG app, after the app has restored previously entered values of static input fields.
update.cotg	Sets the element to the current geolocation.

Image & Annotation

cotgNoteOnImage()

Initializing a new Image & Annotation element prepares it for user interaction.

Example: `$('#myNoteOnImage').cotgNoteOnImage();`

Note

To use this element with a Camera widget instead of a static image, you only have to initialize the Camera widget with the `allowannotations` option set to `true`. The Camera widget will automatically initialize the Image & Annotation widget.

The Image & Annotation element **listens for** the following events.

Event	Description
bind-to-image.cotg	Bind an annotation to the picture.
clear.cotg	Removes any annotations.
clear-note.cotg	Removes any annotations. When using this element with a Camera element, trigger this event to remove annotations without removing the picture.
restore-state.cotg	Restores the annotations when the form is reopened in the COTG app, after the app has restored previously entered values of static input fields.
save-state.cotg	Saves the annotations.

The Image & Annotation element **broadcasts** the following events.

Event	Description
set.cotg	Fired after an annotation has been made.

Locale

cotgLocale()

Initializing a new Locale element sets it to the device's locale.

Example: `$('#myLocale').cotgLocale();`

Events

The Locale element **listens** for the following event.

Event	Description
clear.cotg	Removes the Locale data.

The Locale element **broadcasts** the following event.

Event	Description
set.cotg	This event is fired during initialization of the element, after setting its value to the current locale.

Repository ID

cotgRepositoryId()

Initializing a new Repository ID puts the current Repository ID in the hidden input of this element. The Repository ID is based on the currently logged on COTG user.

Example: `$('#myRepID').cotgRepositoryId();`

Events

The Repository ID element **listens** for the following custom events.

Event	Description
clear.cotg	Removes the Repository ID data.

The Repository ID element **broadcasts** the following event.

Event	Description
set.cotg	This event is fired during initialization of the element, after setting its value to the current Repository ID.

Signature

cotgSignature()

Initializing a new Signature element prepares it for user interaction.

Example: `$('#mySignature').cotgSignature();`

Events

The Signature element **listens** for the following custom events.

Field	Description
clear.cotg	Removes the signature drawing and data.
draw.cotg	Draws the signature on the form (e.g. after a <code>set.cotg</code> or <code>restore-state.cotg</code> event).
restore-state.cotg	Called when the form is reopened in the COTG app, after the app has restored previously entered values of static input fields.
save-state.cotg	Saves the signature data to the local storage of the COTG app.
set.cotg	Sets the given signature. The signature should be given as an SVG string, for example: <pre>\$("#signature").trigger("set.cotg", "<svg>...</svg>");</pre>

The Signature element **broadcasts** the following events.

Field	Description
set.cotg	This event is fired after a user has entered a signature and clicked the Done button of the Signature window on the device.
clear.cotg	Fired after a user has clicked the Done button of the Signature window without entering a signature.
draw.cotg	Fired each time the signature is drawn on the form (by the app, not by the user; e.g. after a <code>set.cotg</code> or <code>restore-state.cotg</code> event).

Time and Formatted Time

`cotgTimePicker()`

Initializing a new (Formatted) Time element prepares it for user interaction.

Example: `$('#myTimePicker').cotgTimePicker();`

Note that the difference between a Time and a Formatted Time is laid down in the HTML structure of the element.

Events

The Time and Formatted Time elements **listen for** the following custom events.

Event	Description
set.cotg	Stores the given time (specified in a Date object).
clear.cotg	Removes the set time.
show-time-picker.cotg	Opens the Time Picker. If no time is provided (specified in a Date object), the current time will be shown.

User Account

cotgUserAccount()

Initializing a new User Account element puts the account of the current user in the hidden input of this element.

Example: `$('#myUser').cotgUserAccount();`

Events

The User Account element **listens for** the following custom event.

Event	Description
clear.cotg	Removes the User Account data.

The User Account element **broadcasts** the following event.

Event	Description
set.cotg	This event is fired during initialization of the element, after setting its value to the current user account.

Content elements

Once you have created template, it can be filled with all kinds of elements, from text to barcodes and from tables to fields on a web form. All types of elements are listed on this page.

There are several ways to insert elements, see "Inserting an element" on page 468.

Each element can have an ID and a class, as well as a number of other properties, depending on the element's type. When an element is selected, its properties can be changed; see "Selecting an element" on page 469, "Attributes" on page 467 and "Styling and formatting an element" on page 470.

ID's and classes are particularly useful with regard to variable data (see "Personalizing Content" on page 592) and styling (see "Styling templates with CSS files" on page 553).

When you add elements, such as text, images or a table, to the content of a template, you are actually constructing an HTML file. It is possible to edit the source of the HTML file directly in the Designer; see "Editing HTML" on the facing page.

Element types

The following types of content can be added to the content of a template:

- "Images" on page 537 and "Dynamic Images" on page 617
- "Text and special characters" on page 546
- "Date" on page 526
- "Table" on page 542 and "Dynamic table" on page 618
- "Boxes" on page 513: Positioned Box, Inline Box, Div and Span

Tip

Wrapping elements in a box (see "Boxes" on page 513) or in a semantic HTML element makes it easier to target them in a script or in a style sheet. Place the cursor in the element or select multiple elements. Then, on the menu, click **Insert > Wrap in Box**. You can now use the wrapper element as a script's or style's

`selector`; see "Using the Text Script Wizard" on page 607 and "Styling and formatting" on page 551.

- "Hyperlink and mailto link" on page 536
- "Barcode" on page 470
- Web "Forms" on page 527 and Web "Form Elements" on page 532
- "Whitespace elements: using optional space at the end of the last page" on page 344 (Print context only)
- "Page numbers " on page 345 (Print context only)
- Article, Section, Header, Footer, Nav and Aside are HTML5 semantic elements; see http://www.w3schools.com/html/html5_semantic_elements.asp
- Other HTML elements: Heading, Address and Pre
- "Snippets" on page 548: a Snippet is a small, ready-to-use piece of content in a file
- Business graphics

Most elements are suitable for use in all contexts. There are a few exceptions, however. Forms and Form elements can be used on web pages only, whereas Whitespace elements and Page numbers can only be used in a Print context. Positioned boxes are well suited for Print sections, but are to be avoided in the Email and Web context.

Whether it is best to use a Table or Box to position text, images and other elements, depends on the context in which they are used; see "How to position elements" on page 567 for more information.

Editing HTML

When you add elements, such as text, images or a table, to the content of a template, you are actually constructing an HTML file.

To see this, toggle to the **Design** tab in the workspace. Click anywhere in the content. Take a look at the *breadcrumbs* at the top of the workspace. The breadcrumbs show the HTML tag of the clicked element, as well as the HTML tags of other elements to which the clicked element belongs. The clicked element is at the end of the line.

To edit the HTML text directly:

- In the workspace, toggle to the **Source** tab.

On this tab you can view and edit the content of the template in the form of plain text with HTML tags (note the angle brackets: <>). You may add and edit the text and the HTML tags, classes, ID's and other attributes.

To learn more about HTML, see for example <https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Introduction> and <http://www.w3schools.com/html/default.asp>.

Many video courses and hands-on courses about HTML (and CSS) are offered on the Internet as well, some for free. Go, for example, to www.codeschool.com or www.codecademy.com and look for HTML (and CSS) courses.

Attributes

ID and class

Every element in the content of a template can have an **ID** and a **class**. ID's and classes are particularly useful with regard to variable data (see "Personalizing Content" on page 592) and styling (see "Styling templates with CSS files" on page 553).

You can specify an ID and/or class when you add the element to the content.

To add an ID and/or class to an element that has already been added to a template, select the element (see "Selecting an element" on page 469) and type an ID and/or a class in the respective fields on the **Attributes** pane at the top right.

Other attributes

Apart from the ID and class, elements can have a varying number of properties, or 'attributes' as they're called in HTML (see "Editing HTML" on the previous page). Which properties an element has, depends on the element itself. An image, for example, has at least four attributes: `src` (the image's URL), `alt` (alternate text), `width` and `height`. These attributes are visible on the **Attributes** pane when you click an image in the content.

For each type of element, a small selection of attributes is visible on the **Attributes** pane at the top right.

Changing attributes via script

Many attributes can be changed via the user interface. Another way to change attributes is by using a script.

Any of the Script Wizards can produce a script that changes an attribute of an HTML element. Set the **Options** in the Script Wizard to **Attribute**, to output the script's results to the value of a specific attribute. See "Using the Text Script Wizard" on page 607.

In code, you can change an element's attribute using the function `attr()`; see "Writing your own scripts" on page 624 and "Designer Script API" on page 874.

Inserting an element

To insert an element in a section:

1. Click the respective toolbar button. Alternatively, click the element on the **Insert** menu.
2. Add an ID and/or a class. ID's and classes are particularly useful with regard to variable data (see "Personalizing Content" on page 592) and styling (see "Styling templates with CSS files" on page 553).
3. Use the **Location** drop-down (if available) to select where to insert the element.
 - **At cursor position** inserts it where the cursor is located in the template.
 - **Before element** inserts it before the HTML element in which the cursor is currently located. For example if the cursor is within a paragraph, the insertion point will be before the `<p>` tag.*
 - **After start tag** inserts it within the current HTML element, at the beginning, just after the start tag.*
 - **Before end tag** inserts it within the current HTML element, at the end, just before the end tag.*
 - **After element** inserts it after the element in which the cursor is currently located. For example if the cursor is within a paragraph, the insertion point will be after the end tag of the paragraph (`</p>`).*

* If the current element is located inside another element, use the **Elements** drop-down to select which element is used for the insertion location. The list displays every element in the breadcrumbs, from the current selection point until the root of the body.

For a list of links to the different types of elements, see "Element types" on page 465.

Selecting an element

When an element is selected, the **Attributes** pane shows the attributes of that element, and the **Styles** pane, next to the **Attributes** pane, shows which styles are applied to it.

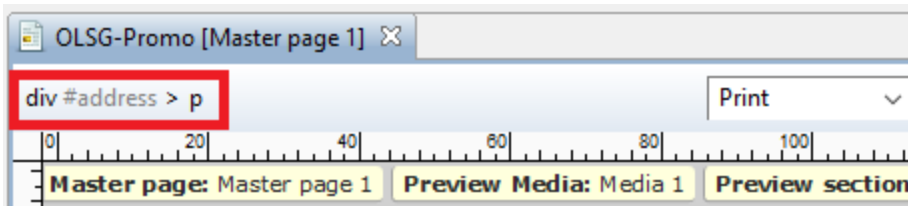
To select an element in the content, you can of course click on it, but this isn't always as easy as it seems, especially when the element has elements inside it.

Tip

Click the **Edges** button on the toolbar to make borders of elements visible on the Design tab. The borders will not be visible on the Preview tab.

There are two more ways to select an element in the content:

- Using the *Breadcrumbs* at the top of the workspace.



Breadcrumbs show the HTML tag of the clicked element, as well as the HTML tags of 'parent elements': elements inside of which the clicked element is located. The clicked element is at the end of the line.

Elements with classes or IDs show these details next to them, for instance `div #contents > ol.salesitems > li`. Click any of the elements in the Breadcrumbs to select that element.

If an element is selected in the Breadcrumbs and the Backspace key is pressed, that element is deleted.

- Using the **Outline** pane. You can find this pane next to the **Resources** pane. It displays a tree view of the elements in the file. Click an element in the tree view to select it.

Deleting an element

To delete an element, select it - as described above - and press the Delete key.

If the deleted element was targeted by a script, you will be asked if you want to delete the script

as well.

Scripts are used to personalize templates. To start learning more about scripts, see "Personalizing Content" on page 592 and "Writing your own scripts" on page 624.

Styling and formatting an element

Format elements directly

Images and other graphical elements can be resized by clicking on them and dragging the resize handles. There are toolbar buttons to color, indent or style text. Other toolbar buttons can left-align, right-align, or rotate graphical elements.

The toolbar buttons only represent a selection of the formatting options for each element. There are no toolbar buttons to change an element's margins, or to add a border to it, for example. To access all formatting properties of an element, you have to open the Formatting dialog. There are two ways to do this:

- Right-click the element and select the type of element on the shortcut menu.
- Select the element (see "Selecting an element" on the previous page) and select the type of element on the **Format** menu.

See "Styling and formatting" on page 551 for more information about the formatting options.

Format elements via Cascading Style Sheets (CSS)

It is highly recommended to use style sheets in templates right from the start. Even more so if the communications are going to be output to different output channels, or if they consist of different sections (for example, a covering letter followed by a policy). Using CSS with templates allows a consistent look and feel to be applied. A style sheet can change the look of multiple elements, making it unnecessary to format each and every element in the template, time and again, when the company's layout preferences change. See "Styling templates with CSS files" on page 553.

Barcode

In PlanetPress Connect Designer, you can add a variety of barcodes to your template. The supported Barcode types include 1d barcodes (the striped ones) and 2d barcodes (encoded horizontally and vertically).

Adding a Barcode

Note

When generating Print output, you can add extra barcodes and OMR marks. The reason why you would do this, is that at merge time more information is available about the actual output document. The page count, for example, is not available at design time.

To add barcodes and OMR marks on the fly when generating Print output, select **File > Print** and check the option **Add additional content** (see "Additional Content" on page 799) in the Print Wizard. To have this done automatically, save this and other output options in an Output Creation Preset: select **File > Print presets > Output Creation Settings** (see "Output Creation Settings" on page 850) and send the preset to Workflow.

Before adding a Barcode, load data or at least a Data Model; see "Loading data" on page 594. You will need the field names when adding the Barcode. Then, to add a Barcode to a section, Master Page or snippet:

1. Select **Insert > Barcode** on the menu or click the **Barcode** toolbar button
2. Choose the desired barcode type. The list is divided between 1d and 2d barcodes.
3. An **ID** is required. You can change the given ID and, optionally, add a class.
4. Check the option **Absolute** to insert the barcode in an absolute-positioned box inside the `<body>` of the HTML, but outside other elements. Alternatively, use the **Location** drop-down to select where to insert the Barcode.
 - **At cursor position** inserts it where the cursor is located in the template.
 - **Before element** inserts it before the HTML element in which the cursor is currently located. For example if the cursor is within a paragraph, the insertion point will be before the `<p>` tag.*
 - **After start tag** inserts it within the current HTML element, at the beginning, just after the start tag.*
 - **Before end tag** inserts it within the current HTML element, at the end, just before the end tag.*
 - **After element** inserts it after the element in which the cursor is currently located. For example if the cursor is within a paragraph, the insertion point will be after the end tag of the paragraph (`</p>`).*

* If the current element is located inside another element, use the **Elements** drop-down to select which element is used for the insertion location. The list displays every element in the breadcrumbs, from the current selection point until the root of the body.

5. Under **Script**, select the field that contains the barcode value. The barcode type dictates the length and exact format of the required value. For a detailed description or for background information on a specific barcode, please refer to the documentation provided by the individual barcode supplier. Note that some barcode readers may require specific parameters as well.

If it is necessary to concatenate fields to compose the barcode value, edit the script after adding the barcode; see "Barcode script" below.

Note

For barcodes that require a Checksum, the Designer can calculate a Checksum if that isn't provided by your data. In that case the field should contain the required value minus the Checksum. To include a calculated Checksum in the barcode value, edit the barcode properties after adding the barcode to the template; see below.

6. Click **OK** to close the dialog.

In the template the barcode shows up as a gray box. The associated barcode script is added to the Scripts pane. To see the barcode script working, toggle to the **Preview** tab in the Workspace.

A barcode is always added with the barcode type's default properties and dimensions, but they can easily be changed; see "Barcode type and properties" on the next page.

Changing a barcode

Barcode script

The barcode script determines which value is fed to the barcode generator. Double-click the script on the **Scripts** pane to change which field or fields are added to the barcode value. When you select more than one field, the script puts the values of the selected fields in one string and passes that to the barcode generator.

Tip

If you don't know which script matches the barcode, click the box that contains the barcode and check the ID of that box on the Attributes pane. Then take a look at the Scripts pane: the selector of the associated script is the same as the ID of the barcode box.

Barcode type and properties

A barcode is always added with the barcode type's default properties and dimensions.

To change the barcode type or the barcode's properties such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select **Barcode** on the shortcut menu.

The barcode properties set via the properties dialog are written to the `data-params` attribute on the barcode element in JSON format. (To see this, select the barcode and open the document in the Source view.)

Click the barcode type below for information about its properties.

- "Code 11, Code 93, Code 93 extended, Industrial 2 of 5, Interleaved 2 of 5, Matrix 2 of 5" on page 497
- "Code 39, Code 39 extended" on page 480
- "UPC-A, UPC-E, EAN-8, EAN-13" on page 511
- "OneCode, KIX Code, Royal Mail, Australia Post" on page 508
- "Code 128" on page 483
- "GS1 DataMatrix" on page 489
- "GS1-128" on page 490
- "Codabar" on page 477
- "MSI" on page 500
- "IMPB" on page 493
- "Postnet" on page 504
- "QR Code" on page 506
- "Data Matrix" on page 485
- "Royal Mail Mailmark" on page 510

- "PDF417" on page 503
- "Aztec Code" on the next page
- "MaxiCode" on page 499

OneCode, KIX Code, Royal Mail, Australia Post

OneCode, KIX Code, Royal Mail and Australia Post are some of the types of barcodes that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode types OneCode, KIX Code, Royal Mail and Australia Post. For the properties of other barcode types, see "Barcode type and properties" on the previous page.

Height, width and spacing

The height, width and spacing of the barcode are all measured in pixels (38 dpi).

- **Bar height:** the height of the (shorter) bars
- **Extended bar height:** the total height of the extended bars
- **Bar width:** the width of the bars
- **Spacing:** the distance between the bars

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.

- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG:** Vector format. This is smaller in size, but not compatible with Email output.
- **PNG:** Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Aztec Code

Aztec is one of the types of barcodes that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode type Aztec. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module size

Enter the size of the square modules in pixels

Configuration type

Use the drop-down to select the format type used when creating the barcode: only full range format, only compact formats, or any format.

Preferred configuration

Use the drop-down to select the preferred format for the barcode. Note that the barcode generator may choose a different format if the data cannot be represented by the preferred format.

Encoding

Use the drop-down to select the encoding type:

- **Normal** can encode any character but is not very efficient for encoding binary values (above 128)
- **Binary** is to be used only if the data contains many bytes/characters above 128.

Error Correction Level

This option reserves a percentage of the symbol capacity for error correction. The recommended percentage for this type of barcode is 23.

Rune

When set to a value between 0 and 255, an Aztec Rune corresponding to the selected value is created. Set the Rune to -1 to disable this feature.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None**: The barcode is rendered based on the module width.
- **Fit to box**: The barcode is stretched to fit the parent box in both width and height.
- **Proportionally**: The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG**: Vector format. This is smaller in size, but not compatible with Email output.
- **PNG**: Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Codabar

Codabar is one of the barcode types that can be added to a template.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the Codabar barcode. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Start Char and Stop Char

Use the drop-down to select the start and stop character for the barcode, which defines the encoding mode. Available characters are A, B, C.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Add Checksum

When checked, PlanetPress Connect will calculate a Checksum character and add that to the result of the Barcode script. If the value to be encoded is longer than 10 digits, a second check character will be calculated.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG:** Vector format. This is smaller in size, but not compatible with Email output.
- **PNG:** Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Code 11, Code 93, Code 93 extended, Industrial 2 of 5, Interleaved 2 of 5, Matrix 2 of 5

Code 11, Code 93, Code 93 extended, Industrial 2 of 5, Interleaved 2 of 5, and Matrix 2 of 5 are a few of the barcode types that can be added to a template.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the following barcode types :

- Code 11
- Code 93
- Code 93 extended
- Industrial 2 of 5
- Interleaved 2 of 5
- Matrix 2 of 5

For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Add Checksum

When checked, PlanetPress Connect will calculate a Checksum character and add that to the result of the Barcode script. If the value to be encoded is longer than 10 digits, a second check

character will be calculated.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG**: Vector format. This is smaller in size, but not compatible with Email output.
- **PNG**: Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Code 39, Code 39 extended

Code 39 and Code 39 extended are two of the barcode types that can be added to a template.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode types Code 39 and Code 39 extended. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Inter Character Gap

Two adjacent characters are separated by an inter-character gap. A value of 1 means that the separator will have the same length as the width of the narrow bars (in centimeters).

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Add Checksum

When checked, PlanetPress Connect will calculate a Checksum character and add that to the result of the Barcode script. If the value to be encoded is longer than 10 digits, a second check character will be calculated.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG:** Vector format. This is smaller in size, but not compatible with Email output.
- **PNG:** Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Code 11, Code 93, Code 93 extended, Industrial 2 of 5, Interleaved 2 of 5, Matrix 2 of 5

Code 11, Code 93, Code 93 extended, Industrial 2 of 5, Interleaved 2 of 5, and Matrix 2 of 5 are a few of the barcode types that can be added to a template.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the following barcode types :

- Code 11
- Code 93
- Code 93 extended
- Industrial 2 of 5
- Interleaved 2 of 5
- Matrix 2 of 5

For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Add Checksum

When checked, PlanetPress Connect will calculate a Checksum character and add that to the result of the Barcode script. If the value to be encoded is longer than 10 digits, a second check character will be calculated.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG:** Vector format. This is smaller in size, but not compatible with Email output.
- **PNG:** Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Code 128

Code 128 is one of the types of barcodes that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode type Code 128. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Code set

Set of characters to be used:

- **A:** ASCII characters 00 to 95 (0–9, A–Z and control codes), special characters, and FNC 1–4
- **B:** ASCII characters 32 to 127 (0–9, A–Z, a–z), special characters, and FNC 1–4
- **C:** 00–99 (encodes each two digits with one code) and FNC 1

In Auto mode, the barcode generator will automatically select the correct encoding mode (set A, B or C) according to the input data.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Add Checksum

When checked, PlanetPress Connect will calculate a Checksum character and add that to the result of the Barcode script. If the value to be encoded is longer than 10 digits, a second check

character will be calculated.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG**: Vector format. This is smaller in size, but not compatible with Email output.
- **PNG**: Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Data Matrix

Data Matrix is one of the types of barcodes that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the QR barcode. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Hex Input

For optimized mailings, German Post requires the supplied data for the Data Matrix barcode to be hexadecimal input.

Check this option if your input data is a hexadecimal code. The incoming data will be interpreted as hexadecimal input and decoded to ASCII before passing the string to the Barcode library.

Dots Per Pixels

Type the number of dots per pixel. To optimize barcode quality a Data Matrix symbol should not be printed with dots smaller than 4 pixels.

Encoding

The data represented in the symbol can be compressed using of the following algorithms.

- **ASCII** is used to encode data that mainly contains ascii characters (0-127)
- **C40** is used to encode data that mainly contains numbers and uppercase characters.
- **Text** is used to encode data that mainly contains numbers and lowercase
- **Base256** is used to encode 8 bit values
- **Auto Detect** automatically detects the data content and encodes using the most appropriate method.
- **None** does not use any encoding.

Preferred format

Use the drop-down to select the size of the Data Matrix.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None**: The barcode is rendered based on the module width.
- **Fit to box**: The barcode is stretched to fit the parent box in both width and height.
- **Proportionally**: The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG**: Vector format. This is smaller in size, but not compatible with Email output.
- **PNG**: Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

UPC-A, UPC-E, EAN-8, EAN-13

UPC-A, UPC-E, EAN-8 and EAN-13 are a few of the barcode types that can be added to a template.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode types UPC-A, UPC-E, EAN-8 and EAN-13. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Show guardbars

Checking this option adds guardbars to the barcode. Guardbars are bars at the start, in the middle and at the end that help the barcode scanner to scan the barcode correctly.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Supplement

UPC-A, UPC-E, EAN-13, and EAN-8 may all include an additional barcode to the right of the main barcode.

- **Type:** The supplement type can be 2-digit (originally used to indicate the edition of a magazine or periodical) or 5-digit (used to indicate the suggested retail price for books). In case this option is set to None, and the data includes digits for the 2 or 5 supplement, the supplement data will be skipped and the additional barcode will not be rendered.

Note

When the chosen supplement type doesn't match the data, the supplement data will be skipped and the additional barcode will not be rendered.

- **Height Factor:** This is the relative height of the supplement's bars compared to the normal bars.
- **Space Before :** Defines the space between the main symbol and the supplement, in cm.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example

[w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG:** Vector format. This is smaller in size, but not compatible with Email output.
- **PNG:** Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

GS1 DataMatrix

GS1 DataMatrix is one of the types of barcodes that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode type GS1 DataMatrix. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Preferred format

Use the drop-down to select the size of the Data Matrix.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.

Add Checksum

When checked, PlanetPress Connect will calculate a Checksum character and add that to the result of the Barcode script. If the value to be encoded is longer than 10 digits, a second check character will be calculated.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG:** Vector format. This is smaller in size, but not compatible with Email output.
- **PNG:** Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

GS1-128

GS1-128 is one of the types of barcodes that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode type GS1-128. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Add Checksum

When checked, PlanetPress Connect will calculate a Checksum character and add that to the result of the Barcode script. If the value to be encoded is longer than 10 digits, a second check character will be calculated.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG:** Vector format. This is smaller in size, but not compatible with Email output.
- **PNG:** Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

OneCode, KIX Code, Royal Mail, Australia Post

OneCode, KIX Code, Royal Mail and Australia Post are some of the types of barcodes that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode types OneCode, KIX Code, Royal Mail and Australia Post. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Height, width and spacing

The height, width and spacing of the barcode are all measured in pixels (38 dpi).

- **Bar height:** the height of the (shorter) bars
- **Extended bar height:** the total height of the extended bars
- **Bar width:** the width of the bars
- **Spacing:** the distance between the bars

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.

- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG:** Vector format. This is smaller in size, but not compatible with Email output.
- **PNG:** Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

IMPB

IMPB is one of the barcode types that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode type IMPB. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG**: Vector format. This is smaller in size, but not compatible with Email output.
- **PNG**: Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Code 11, Code 93, Code 93 extended, Industrial 2 of 5, Interleaved 2 of 5, Matrix 2 of 5

Code 11, Code 93, Code 93 extended, Industrial 2 of 5, Interleaved 2 of 5, and Matrix 2 of 5 are a few of the barcode types that can be added to a template.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the following barcode types :

- Code 11
- Code 93
- Code 93 extended
- Industrial 2 of 5
- Interleaved 2 of 5
- Matrix 2 of 5

For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Add Checksum

When checked, PlanetPress Connect will calculate a Checksum character and add that to the result of the Barcode script. If the value to be encoded is longer than 10 digits, a second check character will be calculated.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG:** Vector format. This is smaller in size, but not compatible with Email output.
- **PNG:** Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Code 11, Code 93, Code 93 extended, Industrial 2 of 5, Interleaved 2 of 5, Matrix 2 of 5

Code 11, Code 93, Code 93 extended, Industrial 2 of 5, Interleaved 2 of 5, and Matrix 2 of 5 are a few of the barcode types that can be added to a template.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the following barcode types :

- Code 11
- Code 93
- Code 93 extended
- Industrial 2 of 5
- Interleaved 2 of 5
- Matrix 2 of 5

For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Add Checksum

When checked, PlanetPress Connect will calculate a Checksum character and add that to the result of the Barcode script. If the value to be encoded is longer than 10 digits, a second check

character will be calculated.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG**: Vector format. This is smaller in size, but not compatible with Email output.
- **PNG**: Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Code 11, Code 93, Code 93 extended, Industrial 2 of 5, Interleaved 2 of 5, Matrix 2 of 5

Code 11, Code 93, Code 93 extended, Industrial 2 of 5, Interleaved 2 of 5, and Matrix 2 of 5 are a few of the barcode types that can be added to a template.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the following barcode types :

- Code 11
- Code 93

- Code 93 extended
- Industrial 2 of 5
- Interleaved 2 of 5
- Matrix 2 of 5

For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Add Checksum

When checked, PlanetPress Connect will calculate a Checksum character and add that to the result of the Barcode script. If the value to be encoded is longer than 10 digits, a second check character will be calculated.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example w3school's color picker).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG**: Vector format. This is smaller in size, but not compatible with Email output.
- **PNG**: Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

MaxiCode

MaxiCode is one of the barcode types that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the MaxiCode barcode. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Resolution

Select the printer output definition for the barcode (200, 300, 400, 500 or 600 dpi).

Mode

PlanetPress Connect supports several modes; for an explanation of these modes see the [MaxiCode page on Wikipedia](#).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

MSI

MSI is one of the types of barcodes that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode type MSI. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Add Checksum

When checked, PlanetPress Connect will calculate a Checksum character and add that to the result of the Barcode script. If the value to be encoded is longer than 10 digits, a second check character will be calculated.

Checksum Type

The Checksum type can be MSI10, MSI11, MSI1010 or MSI1110; see https://en.wikipedia.org/wiki/MSI_Barcode.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG**: Vector format. This is smaller in size, but not compatible with Email output.
- **PNG**: Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

OneCode, KIX Code, Royal Mail, Australia Post

OneCode, KIX Code, Royal Mail and Australia Post are some of the types of barcodes that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode types OneCode, KIX Code, Royal Mail and Australia Post. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Height, width and spacing

The height, width and spacing of the barcode are all measured in pixels (38 dpi).

- **Bar height:** the height of the (shorter) bars
- **Extended bar height:** the total height of the extended bars
- **Bar width:** the width of the bars
- **Spacing:** the distance between the bars

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG:** Vector format. This is smaller in size, but not compatible with Email output.
- **PNG:** Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

PDF417

PDF417 is one of the types of barcodes that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode type PDF417. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Mode

Use the drop-down to set the compaction mode:

- **Binary**: allows any byte value to be encoded
- **Text**: allows all printable ASCII characters to be encoded (values from 32 to 126 and some additional control characters)
- **Numeric**: a more efficient mode for encoding numeric data

Error Correction Level

Use the drop-down to select the built-in error correction method based on Reed-Solomon algorithms. The error correction level is adjustable between level 0 (just error detection) and level 8 (maximum error correction). Recommended error correction levels are between level 2 and 5, but the optimal value depends on the amount of data, printing quality of the PDF417 symbol and decoding capabilities of the scanner.

Nr. of Columns

The number of data columns can vary from 3 to 30.

Nr. of Rows

A PDF417 bar code can have anywhere from 3 to 90 rows.

Bar height

Defines the height of the bars for a single row measured in pixels drawn.

Compact

Check this option to use Compact PDF417 instead of the PDF417 barcode. This shortened form of the PDF417 barcode is useful where the space for the symbol is restricted.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG:** Vector format. This is smaller in size, but not compatible with Email output.
- **PNG:** Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Postnet

Postnet is one of the barcode types that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode type Postnet. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Bar height

You can set the height (in cm) of the short bars and the tall bars in the Postnet barcode.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG**: Vector format. This is smaller in size, but not compatible with Email output.
- **PNG**: Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

QR Code

A QR Code is one of the types of barcodes that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Instead of using the Script wizard (see "Adding a Barcode" on page 471) you could write your own script to get the data for the QR Code; see this how-to: [QR Codes in Designer](#).

Barcode properties

This topic lists the properties of the QR barcode. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module size

Enter the size of the square modules in pixels.

Auto configure

When this option is checked, the barcode generator overwrites the selected Preferred version (see below) and defines the barcode version based on the supplied data.

Preferred version

There are 40 sizes of QR codes. Select the preferred version for the QR code.

Encoding

This option defines the encoding of the barcode. When **Auto** is selected, the barcode generator determines the encoding based on the supplied string. The other options are:

- **Numeric:** 10 bits per 3 digits, with a maximum of 7089 numerical characters.
- **Alphanumeric:** 11 bits per 2 characters, with a maximum of 4296 alphanumeric characters.
- **Byte:** 8 bits per character, with a maximum of 2953 characters.
- **Kanji:** 13 bits per character, with a maximum of 1817 characters.

Extended Channel Interpretation (ECI)

This setting enables data using character sets other than the default set. Select **Latin-1**, **Shift JIS** or **UTF-8**, or select **None** to disable extended channel interpretation.

Correction level

Part of the robustness of QR codes in the physical environment is their ability to sustain 'damage' and continue to function even when a part of the QR code image is obscured, defaced or removed. A higher correction level duplicates data within the QR Code to that effect, making it larger.

FNC

Use the drop-down to either disable FNC or select a FNC option:

- **First:** This mode indicator identifies symbols encoding data formatted according to the UCC/EAN Application Identifiers
- **Second:** This mode indicator identifies symbols formatted in accordance with specific industry or application specifications previously agreed with AIM International. You must then set a value for the Application Indicator property.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.

- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG:** Vector format. This is smaller in size, but not compatible with Email output.
- **PNG:** Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Barcode Data

QR Codes can have many different types of data, which determines how the code will be generated. On top of just straightforward data, special data structures are used to trigger actions on the device that reads them. This can include contact cards, phone numbers, URLs, emails, etc.

To learn more about the specifications of the different QR code types, see the ZXing Project [barcode contents](#) page.

OneCode, KIX Code, Royal Mail, Australia Post

OneCode, KIX Code, Royal Mail and Australia Post are some of the types of barcodes that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode types OneCode, KIX Code, Royal Mail and Australia Post. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Height, width and spacing

The height, width and spacing of the barcode are all measured in pixels (38 dpi).

- **Bar height:** the height of the (shorter) bars
- **Extended bar height:** the total height of the extended bars
- **Bar width:** the width of the bars
- **Spacing:** the distance between the bars

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG:** Vector format. This is smaller in size, but not compatible with Email output.
- **PNG:** Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Royal Mail Mailmark

Royal Mail Mailmark is one of the types of barcodes that can be added to a template; see "Barcode" on page 470.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode type Royal Mail Mailmark. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

The recommendation is to print these barcodes with a module size of 0.5 mm, which equates to 6 dots when printed at 300dpi. The maximum module size for printing is 0.7 mm.

Preferred version

Use the drop-down to select the size of the barcode, in a number of modules. The actual size of the barcode can be 12 mm x 12 mm up to 22.4 mm x 22.4 mm, depending on the preferred version and the module width.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example [w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG**: Vector format. This is smaller in size, but not compatible with Email output.
- **PNG**: Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

UPC-A, UPC-E, EAN-8, EAN-13

UPC-A, UPC-E, EAN-8 and EAN-13 are a few of the barcode types that can be added to a template.

The barcode can be added either using the Barcode toolbar button or through selecting **Insert > Barcode** on the menu; see "Adding a Barcode" on page 471.

Initially the barcode will have the barcode type's default properties. To change those properties, such as the scale and color, open the Barcode properties dialog: right-click the barcode (on the **Design** tab in the Workspace) and select the barcode type on the shortcut menu.

Barcode properties

This topic lists the properties of the barcode types UPC-A, UPC-E, EAN-8 and EAN-13. For the properties of other barcode types, see "Barcode type and properties" on page 473.

Module width

Specifies the width of the narrow bars in centimeters. Changing this value to a higher value will make the barcode bigger when Scale is set to None.

Show guardbars

Checking this option adds guardbars to the barcode. Guardbars are bars at the start, in the middle and at the end that help the barcode scanner to scan the barcode correctly.

Scale

Defines if and how the rendered barcode is scaled in relation to the parent element:

- **None:** The barcode is rendered based on the module width.
- **Fit to box:** The barcode is stretched to fit the parent box in both width and height.
- **Proportionally:** The barcode is stretched up to where it fits either the width or height of the parent box, whichever requires the less stretching.

Supplement

UPC-A, UPC-E, EAN-13, and EAN-8 may all include an additional barcode to the right of the main barcode.

- **Type:** The supplement type can be 2-digit (originally used to indicate the edition of a magazine or periodical) or 5-digit (used to indicate the suggested retail price for books). In case this option is set to None, and the data includes digits for the 2 or 5 supplement, the supplement data will be skipped and the additional barcode will not be rendered.

Note

When the chosen supplement type doesn't match the data, the supplement data will be skipped and the additional barcode will not be rendered.

- **Height Factor:** This is the relative height of the supplement's bars compared to the normal bars.
- **Space Before :** Defines the space between the main symbol and the supplement, in cm.

Human Readable Message

When this option is checked, PlanetPress Connect shows a human readable text below or above the barcode, as defined using the Text Position, using the specified font and font size. The font size is given in points (pt).

Color

The **Color** property allows you to choose a different **Barcode** color (instead of black) and **Background** color (instead of white), by typing a hexadecimal color value (see for example

[w3school's color picker](#)).

Output format

Defines how the barcode is output on the page. There are two possible formats:

- **SVG**: Vector format. This is smaller in size, but not compatible with Email output.
- **PNG**: Binary rasterized format. This is slightly larger than SVG but will display properly in Email output.

Boxes

Boxes are elements that are used to surround other elements, either to style them, to find them, or to place them in specific locations.

Tip

Wrapping elements in a box (or in a semantic HTML element) makes it easier to target them in a script or in a style sheet. Place the cursor in the element or select multiple elements. Then, on the menu, click **Insert > Wrap in Box**. You can now use the wrapper element as a script's or style's selector; see "Using the Text Script Wizard" on page 607 and "Styling and formatting" on page 551.

Tip

With the **Copy fit** feature, text can automatically be scaled to the available space in a Box or Div. See "Copy Fit" on page 566.

Positioned Box

A Positioned Box is one that can be freely moved around the page and does not depend on the position of other elements. A positioned box is actually a `<div>` element that has an **absolute position**; in other words, it has its CSS property `position` set to `absolute`.

Positioned Boxes are suitable for use in Print templates only.

Adding a Positioned Box

To insert a Positioned Box, use the  icon on the toolbar.

Moving and resizing a Positioned Box

Positioned Boxes can be moved by dragging the borders, and resized using the handles on the sides and the corners. Pressing any arrow key moves the box by 1 pixel in the direction of that key.

Alternatively the size and position can be set on the Attributes pane. Note that the size and offset values will be displayed in the default print units as defined in the preferences.

To move or resize multiple Positioned Boxes at the same time, hold the Ctrl key while selecting them. You could either select them in the Design view (the main editor) or in the Outline pane.

Dynamically changing the position

A Positioned Box has the following attributes:

- **anchor** defines the page number (starting by 0) the box is placed on
- **offset-x** defines the horizontal position of the box relative to its container
- **offset-y** defines the vertical position of the box relative to its container.

These attributes can be set in a script. The following script dynamically changes the position of a Positioned Box in a Print context by setting the offset-x and offset-y values.

```
results.attr('offset-x', '96');  
results.attr('offset-y', '96');
```

The measurements are in pixels (e.g. 96px = 1in). Note that you do not need to set the units.

Note

Do not set the `top` or `left` property of a Positioned Box in a style sheet. The position of a Positioned Box in a Print context is handled via its attributes to take the page (or Master Page) and page margins into account. Attributes cannot be overwritten from within a style sheet: style sheets specify style properties, not values of attributes.

Styling a positioned box


A Positioned Box can be styled using the **Format > Box** menu item, through the CTRL+M keyboard shortcut, or through CSS; see "Styling and formatting" on page 551 and "Styling templates with CSS files" on page 553.

Inline Box




An Inline Box is one that is placed within the text flow, where other elements (including text) can wrap around it. An inline box is actually a <div> element that is **floating**; in other words, it has its CSS property `float` set to `left`, `right` or `no float`.

Inline Boxes can be used in Print context and in Web pages. It is common to do entire web layouts using the `float` property. In Email templates, it is best to use Tables to position elements.

Adding an Inline Box

To insert an inline box, use the  icon on the toolbar. Inline Boxes can be resized using the handles on the sides and corner. They can be styled using the **Format > Box** menu item, through the CTRL+M keyboard shortcut or through the CSS files; see "Styling and formatting" on page 551 and "Styling templates with CSS files" on page 553.

Positioning an Inline Box

Initially an Inline Box will float to the left. Use the  (Float left),  (No float) and  (Float right) icons on the toolbar to change the position of an Inline Box within the text.

- The **Float left** button aligns the Inline Box to the left. The text is positioned to the right of it and is wrapped around the box.
- The **Float right** button aligns the Inline Box to the right, with the text wrapped around it to the left.
- The **No float** button positions the Inline Box where it occurs in the text.

It is not possible to move an Inline Box using drag and drop. To move the Inline Box to another position in the text, you have to edit the HTML on the Source tab in the Workspace, moving the <div> element using cut and paste. To open the Source tab, click it (at the bottom of the Workspace) or select **View > Source View**.

Span

The Span element (in HTML code) is used to group inline elements, such as text in a paragraph. A Span doesn't provide any visual change by itself, but it provides a way to target its content in a script or in a style sheet.

To wrap content in a span, select the text and other inline elements and click **Insert > Wrap in Span** on the menu. Give the span an ID, if you are going to add a style rule or script for it that is unique to this span; or give the span a class, if this span can be targeted by a style or script along with other pieces of content. Now you can use the wrapper's ID or class as a script's or style's `selector`; see "Using the Text Script Wizard" on page 607 and "Styling and formatting" on page 551.

Div

The Div is the element used to create both Positioned Boxes and Inline Boxes. By default, a Div element reacts pretty much like a paragraph (<p>) or an inline box set to 'no float' except that it can be resized directly. Just like Positioned Boxes and Inline Boxes, Div elements can be styled using the **Format > Box** menu item, through the CTRL+M keyboard shortcut or through the CSS files; see "Styling and formatting" on page 551 and "Styling templates with CSS files" on page 553.

Adding a Div element

To add a Div, select **Insert > Structural Elements > Div** on the menu. For an explanation of the options, see "Inserting an element" on page 468.

HTML tag: div, span

When you add elements, such as text, images or a table, to the content of a template, you are actually constructing an HTML file. It is possible to edit the source of the HTML file directly in the Designer; see "Editing HTML" on page 466.

In HTML, boxes are <div> elements. Spans are elements. To learn how to change the attributes of elements, see "Attributes" on page 467.

Business graphics

Business graphics display variable data, originating from one record, in a graphical way. Three types of business graphics are available: Pie Charts, Bar Charts and Line Charts.

Instead of using one of the pre-defined graphic types you could create your own dynamic chart for use in a Connect Web page, as described in the following how-to: [Dynamic dashboard charts](#).

Adding a business graphic

To add a business graphic to the template:

1. Place the cursor where the graphic should be added.
2. Click the toolbar button of the type of chart you want to add, or select **Insert > Business graphic** and choose the chart type.
3. An **ID** is required. You can change the given ID and, optionally, add a class.
4. Use the **Location** drop-down to select where to insert the graphic:
 - **At cursor position** inserts it where the cursor is located in the template.
 - **Before element** inserts it before the HTML element in which the cursor is currently located. For example if the cursor is within a paragraph, the insertion point will be before the <p> tag.*
 - **After start tag** inserts it within the current HTML element, at the beginning, just after the start tag.*
 - **Before end tag** inserts it within the current HTML element, at the end, just before the end tag.*
 - **After element** inserts it after the element in which the cursor is currently located. For example if the cursor is within a paragraph, the insertion point will be after the end tag of the paragraph (</p>).*

* If the current element is located inside another element, use the **Elements** drop-down to select which element is used for the insertion location. The list displays every element in the breadcrumbs, from the current selection point until the root of the body.

5. Use the **Input format** drop-down to select the source of the data for the Chart:
 - With **Static Labels** the chart has the same number of items for each record. Select the data fields (with a numerical value) and type a label for those fields.

All other options fill the chart dynamically, using data from a detail table. At least one detail table must be available in the Data Model for this option to be functional.

- **Dynamic labels** (Pie Charts only). The chart uses **one color per record** in the detail table. Select the data field under **Values**.

- **Columns are series** (Bar Charts and Line Charts only). The chart uses **one color per data field** selected under **Columns**. For each data field, a series of bars or lines in one colour is added to the chart.
- **Rows are series** (Bar Charts and Line Charts only). The chart uses **one color per record** in the detail table. For each data field selected under **Columns**, a series of bars or lines in different colours is added to the chart.

With the options to fill a chart dynamically you also have to select a **detail table** and a **(row) label**: a data field of which the value appears near the parts in a pie chart or under the bars or points of the line in a bar chart or line chart. The label is also used for the legend. Note that initially the legend is not visible. To make it visible, check the option **Show legend** in the chart's properties (see "Business graphic properties" below).

6. Close the dialog and go in Preview mode in the Workspace to view the result.

Changing a business graphic

Chart script

The script related to a business graphic determines which values are used to generate the graphic. Double-click the script on the **Scripts** pane to change this.

Tip

To find the script that fills the chart, hover over the names of the script in the Scripts pane. That script will highlight the chart in the template and its selector is the same as the ID of the business graphic (preceded by #).

For an explanation of the options in the Script wizard, see **step 5** of "Adding a business graphic" on the previous page.

Business graphic properties

A chart is always added with the chart's default properties. To change those properties, such as the colors and the legend, open the Chart properties dialog: right-click the business graphic (on the **Design** tab in the Workspace) and select the respective business graphic on the shortcut menu.

For an overview of all properties, see:

- "Bar Chart Properties dialog" on page 666
- "Line Chart Properties dialog" on page 683
- "Pie Chart Properties dialog" on page 692.

COTG Elements

With the Designer you can create Capture OnTheGo templates. COTG templates are used to generate forms for the Capture OnTheGo mobile application. This topic is about Capture OnTheGo form elements. For more information about the application refer to these websites: [Capture OnTheGo](#) and [Capture OnTheGo in the Resource Center](#).

Capture OnTheGo (COTG) elements can only be added within a Form element in a Web context; see "COTG Forms" on page 406. For information about how to add and use COTG Elements, see "Using COTG Elements" on page 431.

It is also possible to add COTG Elements dynamically, to set defaults for COTG elements and to react to custom events that occur when a user interacts with a COTG element. For more information see: "Using the COTG plugin: cotg-2.0.0.js" on page 442 and "Dynamically adding COTG widgets" on page 445.

Barcode Scanner

The Barcode Scanner element adds a button to trigger the device to scan a barcode. A very large variety of barcode types are supported (see the table below). Once the barcode has been scanned, the data from the barcode will be added to the COTG Form and submitted along with it.

Note

Using the Barcode Scanner element requires the installation of the [ZXing Barcode Scanner](#) application on Android devices. The application returns the barcode data after scanning.

Supported barcodes

Which barcodes are supported depends on the operating system of the device, (and, on iOS, possibly also on the actual version of the system).

Barcode Type	iOS	Android, Windows
Aztec Code	x	x
Codabar		x
Code 39	x	x
Code 93	x	x
Code 128	x	x
Data Matrix	x	x
EAN_8	x	x
EAN_13	x	x
I2OF5	x	
ITF	x	x
PDF417	x	x
QR CODE	x	x
RSS14		x
RSS_EXPANDED		x
UPC_A		x
UPC_E	x	x

Camera

The Camera element adds a group of buttons to capture or select an image. Once the image is selected via the camera or the device's library (aka "gallery"), it is saved within the Form data.

When the form is submitted, the image is sent in a base64-encoded string format. To learn how to add Camera data to a template, see "Adding Camera data to the template" on page 412.

The Camera element has a number of options, of which most can be set in the Design view. These options are described below.

All options, including options that cannot be set via the Design view, can be set via the Source view or in code; see "Changing default settings for widgets" on page 444.

Buttons

By default, the Camera element adds three buttons to the form:

- **Take now:** Opens the device's default Camera application to take a picture using the device's camera. Capture OnTheGo has no impact on the device's applications, so the features available (quality, orientation, filters) are dependent on the device itself. You can, however, set the format, quality and scaling for images that are submitted by the Camera element, as explained below.
- **Library:** Opens the device's default library or gallery application to select a single image that is then saved in the form data. The accessible images and navigation depend on the device itself.
- **Clear:** Removes any existing image data from the Camera element.

To omit the Take now or Library button, edit the Camera element's properties: right-click the Camera element after adding it to the form, select **Camera properties** and then use the **Source** drop-down to select which buttons will be available: Take, Pick from library, or both.

Annotations

Annotations can make a picture much more informative: an arrow, showing in which direction a car was driving; a circle, where the car has been damaged... To allow the user to make annotations, right-click the Camera element after adding it to the form, select **Camera properties**, and then check **Allow annotations**.

Clicking (or rather, touching) the image will bring up the annotation dialog. Annotations can be made in a Marker (semi-transparent) or Pencil (solid) style, in different colors and with different widths.

Annotations are submitted in SVG format by a hidden input added to the Camera element. The name of that input is the ID of the Camera element, followed by "-note-data", for example **camera1-note-data**.

Cropping/editing/deskewing

To allow the user to crop, edit and deskew the image after taking or selecting it, select **Camera properties**, and then check **Edit Image** and/or **Allow Deskew**. Which editing options the user actually gets and how they are presented to the user depends on the operating system of the device. On an Android device for example, the user may be able only to crop the image, while the user of an iOS device may select part of the image and rotate that selection.

Image format

You can set the format, quality and scaling for images that are submitted by the Camera element. Right-click the Camera element after adding it to the form, select **Camera properties** and edit the Image properties:

- **Format:** The image format can be PNG or JPG.
- **Quality:** Set the compression in a percentage.
- **Scale Image:** Check this option to enable image scaling. Then set the maximum width and height of images before they are sent to the server. Note that only the smallest of these is applied and the size ratio is always maintained.

Time stamp

A time stamp can be added to each picture taken. Right-click the Camera element after adding it to the form, select **Camera properties**, and then check **Add Time Stamp**.

The time stamp will be added to the bottom left of the picture, with medium font size, and long date format (for example: 6/15/2009 1:45 PM). These settings can only be changed via the Source tab or in code; see "Using the COTG plugin: cotg-2.0.0.js" on page 442 and the Capture OnTheGo API: "Camera" on page 454.

Note

The Time stamp feature doesn't work in versions of the app prior to 10.6.

How to use the captured or selected image in a template

After a user has submitted the form and the data has been extracted, you may want to display the captured or selected image in a Designer template, for example in a letter or on a web page. To do this:

1. Load the data mapping configuration (or at least the data model).
2. Insert a dummy image in the template.
3. Right-click the dummy image and select **Dynamic Image**. The Text Script Wizard appears.
4. Under **Field** select the field that contains the base64-encoded string. The script puts the given string in the source (**src**) attribute of the image (****).

Instead of using the Text Script Wizard, you could also write a script yourself; see "Writing your own scripts" on page 624.

Date and Formatted Date

The Date element and the Formatted Date element display the current date on the device when the form is first opened. When the element is touched, a date selector appears so the user can modify this date. The Formatted Date element displays dates in a format that depends on the locale of the device on which the user is viewing the form. A Date Element displays dates in the ISO 8601 format: YYYY-MM-DD.

When the form is submitted, the date data is sent as plain text. A Formatted Date element submits the date in two formats: in the format that depends on the device's regional and language settings and in the ISO format mentioned above (using a hidden field). A Date element sends the date in the ISO format only.

Device Info

The Device Info Element adds a field that contains some information about the device (phone or tablet) that is submitting the COTG Form. This includes the device's type (Android or iOS), operating system version, device model and its UUID (Universally Unique Identifier). This information can be useful for both troubleshooting, if errors occur on specific device types for example, as well as for security validation: it is possible to maintain a list of device UUIDs that are allowed access, to prevent unauthorized use even if someone has a user name and password to a repository.

Document ID

The Document ID element retrieves the Document ID of the form currently viewed by the app. You could put the Document ID in a hidden input, so that when the form is submitted, the Document ID is submitted as well. A Document ID can be used on the server side to check (in the Connect database) if the data has already been submitted.

Fields Table

The Fields Table element adds a table with two rows, a delete button at the end of the first row and an add button at the end of the second row. Inside the rows you can put whatever elements you need. The user can click (or rather, touch) the Add button to add a row to the table. The new row will contain the same elements as the first row. The names of all elements in the first row will be extended with `__0`, while the names of the elements in the second row will be extended with `__1`, etc. (This means that the submitted data can be grouped; see "Grouping data using arrays" on page 434.)

Geolocation

The Geolocation Element adds a button to read the device's current GPS coordinates and save them in a form field. When the button is pressed, the GPS coordinates are requested and saved. When the form is submitted, the Geolocation data is sent in plain text.

High accuracy

By default, devices attempt to retrieve a position using network-based methods. To tell the framework to use more accurate methods, such as satellite positioning, the High Accuracy setting has to be enabled on the Geolocation element.

To make this setting, right-click the Geolocation element (or select it on the Outline pane) after adding it to the form, select **Geolocation properties** and check the **High Accuracy** option.

Note

The Geolocation element has several options, of which only one can be set via the user interface. All options, including those that cannot be set in Design view, can be set via the `data-params` attribute in the HTML, or in code; see "Using the COTG plugin: `cotg-2.0.0.js`" on page 442.

Image & Annotation

The Image & Annotation element is meant to be used with an image that needs input from the user. When inserting an Image & Annotation element you have to select the image. The user can simply click (or rather, touch) the image to bring up the annotation dialog. Annotations can be made in a Marker (semi-transparent) or Pencil (solid) style, in different colors and with different widths.

Annotations are submitted in SVG format by a hidden input. The name of that input is the ID of the Image & Annotation element, followed by `"-note-data"`, for example **image_annotation1-note-data**.

Locale

The Locale Element does not have a UI element in the form. Inserting it adds a hidden input field that will contain the device's set locale when the form is submitted. This data is sent in plain text format and is available when processing the form data. The format is defined by the device.

Repository ID

The Repository ID element retrieves the repository ID (read only) from the app based on the currently logged on COTG user. You could put the Repository ID in a hidden input, so that when the form is submitted, the Repository ID is submitted as well. This information can be used on the server side to take specific actions, such as sending properly branded emails.

Signature

The Signature Element adds a signature box to a COTG form. These signatures are filled in via touch input, either with a finger or capacitive pen. Touching the signature box opens up a fullscreen box used to sign (generally more useful in Landscape mode depending on the device); after confirming, the dialog saves the data into the Form.

Signature data is transmitted in SVG plain text format. This type of data can be stored in a data field in a Data Mapping and dragged from the Data Model into a template as is. In Preview mode it will be displayed as an image because the Designer, just like web browsers, knows how to display this kind of data.

Note

The Signature data returned by the COTG app (as of Android 10.6.0, iOS 10.6.0, and Windows 6.0) is formatted so that the signature will automatically be scaled to fit in the containing box in a template. With previous versions of the app, the format of returned signatures could vary.

Time and Formatted Time

The Time element and the Formatted Time element display the current time on the device when the form is first opened. When the element is touched, a time selector appears so the user can modify this time. The Formatted Time element displays times in a format that depends on the locale of the device on which the user is viewing the form. A Time Element displays dates in the ISO 8601 format: HH:MM.

When the form is submitted, the time data is sent as plain text. A Formatted Time element submits the time in both the ISO format mentioned above and in the format that depends on the device's regional and language settings. A Time element sends the time in the ISO format only.

User Account

The User Account Element adds a hidden field that contains the Capture OnTheGo user account (an email address) that was used to submit the form to the server. This is useful if a form is available to many different users, to detect who submitted it.

Date

The Date element inserts the current system date, optionally making it dynamic so that it updates whenever the template is viewed or produces output.

Adding a date

To add a Date element, use the **Insert > Date** option in the menus. A dialog appears with the following controls:

- **Language:** Use the drop-down to select which language the date should be displayed in.
- **Update Automatically:** Check to update the date automatically when the template is viewed or produces output. When this option is checked, a placeholder is inserted in the template and a script is created to update it automatically, otherwise a static text with the date is inserted.
- **Available Formats:** Select the date/time format in which to display the date.

Click OK to insert the date or Cancel to close the dialog.

Tip

If you are looking to add a date that originates from a record set, to a template, see: "Variable Data" on page 604. To insert a date you could use either the drag and drop method or the Text Script Wizard; the latter lets you set the date/time format.

Changing the date

Once inserted, a date can be modified directly in the template (if it does not update automatically) or through the **date** script (if it does update automatically). To modify the date in the script:

1. Double-click the **date** script in the **Scripts** pane.
2. Between the round brackets after Date, enter the desired date in the following order: year, month, day, and optionally hours, minutes, seconds, milliseconds (see http://www.w3schools.com/js/js_dates.asp and https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Date.) When the time is omitted, it defaults to 12:00:00 AM.

Formatting an automatically updating date

The script added to automatically update the date uses the short date format. To change this:

1. Double-click the **date** script in the **Scripts** pane.
2. Delete the first line of the script.
3. On the second line, delete what comes after `format` and change `format` to `formatter` (see "formatter" on page 912).
4. Now type a dot after `formatter`, press **Ctrl + space** and choose one of the functions to format a date and time; see "Creating a Date object from a string" on page 917.

Note

The Locale, set in the **Edit > Locale** dialog, has an influence on the formatting of a date. The Locale can be the system's locale, a specific locale, or it can depend on the value of a data field; see "Locale" on page 590.

Forms

Web templates can contain Forms. Capture OnTheGo templates always contain a Form.

Tip

To create a Capture OnTheGo template, preferably use a Template Wizard (see "Capture OnTheGo

template wizards" on page 416). The Wizard doesn't just add the form, it also adds the necessary Capture OnTheGo form elements (see), style sheets and JavaScript files, and extra pre-made elements.

Adding a Form

This procedure describes how to add a Form element to an existing Web context.

1. On the **Resources** pane, expand the **Web** context and double-click a Web page to open it.
2. To use the Form Wizard, select the **Insert > Form Wizard** menu option. The Form Wizard adds a Form to the Web page including the specified fields.
Alternatively, you can select **Insert > Form** on the menu to open a dialog that lets you set the Form's properties, validation method and location, but doesn't allow you to specify fields. If you choose this method, skip step 8 and 9 of this procedure and add fields after inserting the Form (see "Adding elements to a Form" on page 398).
3. Add an **ID** and/or a **class**. ID's and classes are particularly useful with regard to variable data (see "Personalizing Content" on page 592) and styling (see "Styling templates with CSS files" on page 553).
4. In the **Action** field, enter the URL where the form data should be sent. The URL should be a server-side script that can accept form data. The **action** of a Capture OnTheGo form should specify the Workflow HTTP Server Input task that receives and handles the submitted data. The action will look like this: **http://127.0.0.1:8080/action** (8080 is Workflow's default port number; 'action' should be replaced by the HTTP action of that particular HTTP Server Input task). The **method** of a Capture OnTheGo form should be **POST** to ensure that it doesn't hit a data limit when submitting the form. The GET method adds the data to the URL, and the length of a URL is limited to 2048 characters. Especially forms containing one or more Camera inputs may produce a voluminous data stream that doesn't fit in the URL. GET also leaves data trails in log files, which raises privacy concerns. Therefore POST is the preferred method to use.
5. Using the the **Method** drop-down, select whether the form should be sent using the GET or POST method.
6. Using the next drop-down, select the form's Encryption Type (**enctype**):
 - **application/x-www-form-urlencoded**: Default. All characters are encoded before they are sent. Spaces are converted to "+" symbols, and special characters are

converted to ASCII HEX values.

- **multipart/form-data**: No characters are encoded. This value is required when you are using forms that have a file upload control.
- **text/plain**: Spaces are converted to "+" symbols, but no special characters are encoded.

7. Select a **validation** method:

- The **Browser** validation method leaves it up to the browser to validate the user input. When adding fields to the Form (see the next step) you can only make fields required and set the maximum length as an additional requirement for some fields.
- Select **jQuery Validation** to validate using JQuery scripts. This allows you to specify stricter requirements per field and type a different message for each field to display to the user if the input is not valid. This method ensures a more consistent validation as it is browser independent. The necessary JQuery files will be added to the JavaScript folder on the Resources pane when the form is inserted.

8. Under **Fields**, click the **Add** button and click on the desired field type to add a field of that type; see "Form Elements" on page 532.

Note

A Fieldset is not available in the Form Wizard, because a Fieldset itself can contain multiple different fields. Add the Fieldset after inserting the Form; see "Adding elements to a Form" on page 398.

9. Double-click each field in the Fields list and change its settings. For an explanation of the settings, see "Adding elements to a Form" on page 398.
10. The order of the elements in the list under **Fields** determines in which order the elements will be added to the Form. Use the **Move Up** and **Move Down** buttons to change the order of the elements in the list.
11. Use the **Location** drop-down to select where to insert the element.
- **At cursor position** inserts it where the cursor is located in the template.

- **Before element** inserts it before the HTML element in which the cursor is currently located. For example if the cursor is within a paragraph, the insertion point will be before the <p> tag.*
- **After start tag** inserts it within the current HTML element, at the beginning, just after the start tag.*
- **Before end tag** inserts it within the current HTML element, at the end, just before the end tag.*
- **After element** inserts it after the element in which the cursor is currently located. For example if the cursor is within a paragraph, the insertion point will be after the end tag of the paragraph (</p>).*

* If the current element is located inside another element, use the **Elements** drop-down to select which element is used for the insertion location. The list displays every element in the breadcrumbs, from the current selection point until the root of the body.

12. Close the dialog. Now you can start adding elements to the Form (see "Using Form elements" on page 398, "Form Elements" on page 532, and "COTG Elements" on page 519).

Changing a Form's properties and validation method

Once a Form has been added, you can of course edit its HTML code directly in the Source view of the workspace. Apart from that, there are a number of dialogs to change a Form's properties and validation settings.

Changing a Form's properties

1. Select the form (see "Selecting an element" on page 469).
2. On the **Attributes** pane you can change:
 - The **ID** and/or **class**. ID's and classes are particularly useful with regard to variable data (see "Personalizing Content" on page 592) and styling (see "Styling templates with CSS files" on page 553).
 - An **Action**: the URL where the form data should be sent. The URL should be a server-side script that can accept form data.
 - A **Method**: this defines whether the form should be sent using the GET or POST method.

- An Encryption Type (**enctype**):
 - **application/x-www-form-urlencoded**: Default. All characters are encoded before they are sent. Spaces are converted to "+" symbols, and special characters are converted to ASCII HEX values.
 - **multipart/form-data**: No characters are encoded. This value is required when you are using forms that have a file upload control.
 - **text/plain**: Spaces are converted to "+" symbols, but no special characters are encoded.

Changing a Form's validation method

In Connect PlanetPress Connect, there are two ways in which a Form's input can be validated:

- The **Browser** validation method leaves it up to the browser to validate the user input. When adding fields to the Form (see the next step) you can only make fields required and set the maximum length as an additional requirement for some fields.
- **JQuery Validation** validates input using JQuery scripts. This allows for stricter requirements per field and a different message for each field to display to the user if the input is not valid. This method ensures a more consistent validation, as it is browser independent. The necessary JQuery files will be added to the JavaScript folder on the Resources pane when this option is chosen.

To change a Form's validation method:

1. **Right-click** on the Form element and choose **Validation settings**.
2. Choose a validation type (see above).
3. Double-click each field in the list to edit their validation settings:
 - **Required**: Check if the field is required to submit the form. If a field is required but contains no data, a message will be shown to the user.
 - **Minimum length**: Enter a numerical value for the minimum character length required for this field.
 - **Maximum length**: Enter a numerical value for the maximum character length accepted for this field.
 - **Equal to**: Use the drop-down to select another field that is already added to the same Form. The contents of both fields must match for the data to be validated. This is useful for confirmation fields such as for passwords, email addresses etc.

Which of these options are available depends on the validation method of the form: with

Browser validation you can only make a field required and set a maximum length.

Changing a Form's validation in HTML

In HTML, the validation method is stored in the `data-validation-method` attribute of the `<form>` element, with the value "browser" or "jquery-validation".

A custom message to be shown when validation of a particular Form element has failed, can be stored in the `data-custom-message` attribute of the Form element, for example:

```
<input id="email1" name="email1" data-custom-message="Enter a valid email address." type="email" required="">
```

Validation in Connect 1.0.0

In Connect 1.0.0, the validation method of the template was stored using the names "standard" and "custom". Standard has changed to "browser" and custom is now "jquery-validation".

When you open a template made with that version of the software, the template will be migrated to use the new attribute values for the `data-validation-method` attribute of the `<form>` element. The JavaScript file `web-form-validation.js` will not be migrated: delete that file and then change the Form's validation method to jQuery Validation, as described above. When you click OK, the new version of the `web-form-validation.js` file will be added.

Submitting a Form

When a form is submitted, by clicking or touching the Submit button, the **name** and **value** of form elements are sent to the address that is specified in the Form's action (see "Adding a Form" on page 528 or "Changing a Form's properties" on page 530). If the `name` attribute is omitted, the data of that input field will not be sent at all.

The Form's validation should ensure that the data that the user submits is valid.

Form Elements

This topic lists the elements that can be added to a form in a Web page or in a Capture OnTheGo template and explains how to add them to a Form, set a default value or change their validation. For more information about Forms and Form elements in the Designer, see "Forms" on page 527 and "Using Form elements" on page 398.

For more information about elements in Forms, see http://www.w3schools.com/html/html_forms.asp.

Fieldset

A fieldset is a group of related elements in a form. The elements don't have to be of the same type. After inserting and selecting the Fieldset (see "Selecting an element" on page 469) you can add elements to it in the same way you add elements to a Form; see "Adding elements to a Form" on page 398.

Text

The Text element is a simple `<input>` element with the type `text`. It accepts any alphanumerical characters, including special characters. The string is HTML-encoded before it is submitted to the server.

Email

The Email element is a text input field that accepts only email addresses in a valid format.

URL

The URL element is a text input field that accepts only URLs (a web page address) in a valid format.

Password

The Password element is a password field that accepts any alphanumerical characters. When the user types in this field, characters are shown as asterisks only.

Number

The Number element is a text field accepting only numbers.

Date

The Date element is a text field accepting only dates in a valid format.

Text area

The Text area element is a text field accepting multiple paragraphs. You can set a number of rows when adding the Text Area to the Form, and change it on the Attributes pane.

Hidden field

A hidden field can contain specific data used by the server-side script. It is not visible to the user. When adding a Hidden Field you can set the value that will be sent on submit.

Label

A Label element is a text displaying informative text within the form. Labels are non-interactive. Note that this type of label is not tied to an input element. At the same time you add an input element, you can add a label to that element; see "Adding elements to a Form" on page 398. To change a label after inserting the Form, simply click it and start typing.

Checkbox

A Checkbox element sends information to the server whether it is checked (true) or not (false). When adding a Checkbox you can set its initial state and its value. The contents of the value property do not appear in the user interface. If a Checkbox is in checked state when the form is submitted, the `name` of the checkbox is sent along with its value.

If a Checkbox is not checked, no information is sent when the form is submitted. Fortunately, there is a workaround to submit the status of an unchecked checkbox; see "Using Form elements" on page 398.

Radio Button Group

A Radio Button Group is not an input element itself. Rather it is a group of Radio Buttons that have the same `submit name`, indicating that they belong to the same group. Multiple Radio Buttons in the same group only accept one option to be selected. Only the value of the selected Radio Button will be sent to the server on submitting the form. If more options are to be allowed at the same time you should use Checkboxes instead.

The option to add a Radio Button Group is only available in the Form Wizard; see "Forms" on page 527. You could also create a Radio Button Group by specifying the same submit name for a number of Radio Buttons when adding them to a Form.

Radio Button

A radio button sends information to the server whether it is selected (true) or not (false). When adding a Radio Button you can set its initial state and its value. The contents of the value property do not appear in the user interface. If a Radio Button is in selected state when the form is submitted, the `name` of the Radio Button is sent along with its value.

If a Radio Button is not checked, no information is sent when the form is submitted. Fortunately, there is a workaround to submit the status of the unchecked radio button, see "Using Form elements" on page 398.

The `submit name` of a Radio Button indicates to which Radio Button Group the Radio Button belongs.

Select

A Select element is a drop-down list with multiple entries from which the user can select only one option. When adding a Select to a Form you can type the options to be given in the drop-down list and the values related to them. Only the value of the selected option will be sent to the server on submitting the form.

To learn how to dynamically add options to a Select element, see this how-to: to [Dynamically add options to a dropdown](#).

Button

The Button element is an element that can be clicked. When adding a Button to a Form you can set the button's **type**:

- A **submit** button will validate the form data and if validation is successful, send the data to the provided URL (by the Action specified for the Form; see "Changing a Form's properties" on page 530).
- A **reset** button will reset the form to its original configuration, erasing any information entered and options provided. **Note:** This cannot be undone!

The button's type can also be changed on the Attributes pane.

There may be multiple submit buttons on a Form. If this is the case, specify a different `name` and/or `value` for each of the buttons.

Note: When adding a Button to a Form, you can specify a value, but no name. The Button's ID will be copied to the element's `name` attribute. However, after inserting the Button you can change its `name` on the Attributes pane.

Hyperlink and mailto link

Links can be added to any template but they only work in electronic output (web pages, email and PDF files). They can be a regular hyperlink pointing to a web page or a mailto link that will open the default email client when clicked.

HTML element: a

When you add elements, such as text, images or a table, to the content of a template, you are actually constructing an HTML file. It is possible to edit the source of the HTML file directly in the Designer; see "Editing HTML" on page 466.

The HTML tag of a hyperlink or mailto-link is `<a>`. This is sometimes called an `anchor` tag. For a list of attributes, see http://www.w3schools.com/tags/tag_a.asp.

Adding a hyperlink or mailto link

1. Select text or an image.

Note

Although it is possible, it is not advisable to add a Hyperlink to other elements, such as a Paragraph or Div. HTML 4 specifies that hyperlinks and mailto-links may only contain inline elements. Block elements, such as a Div, may not appear inside a link. HTML 5 states that the link "may be wrapped around entire paragraphs, lists, tables, and so forth, even entire sections, so long as there is no interactive content within (e.g. buttons or other links)"; see <https://www.w3.org/TR/html5/text-level-semantic.html>.

2. Click the **Insert hyperlink** button on the toolbar, or on the menu, select **Format > Hyperlink > Insert**.
3. Select URL to create a regular hyperlink pointing to a web page, or select Email to create a mailto-link that will open the default email client when clicked.
4. For a **URL**:
 - **URL**: enter a valid, well-formed URL to link to. It must start with the protocol, such as `http://` or `https://`.

- **Target:** use the drop-down or type in the target for the link. When the target is `_blank` the link will open in a new browser window or tab.

For a **mailto link**:

- **Email:** enter a valid email address that appears by default in the To: field of the email client.
- **Subject:** type a default subject that appears in the Subject: field of the email client.
- **Message:** type a message that appears by default in the Message field of the email client.

Note that all these can be changed within the email client once the link is clicked.

Dynamically inserting or modifying a hyperlink

You may wish to adjust a hyperlink depending on a value in a record that is merged to the template when generating output, for example, to provide a different mailto link for different customers.

How to add or modify a hyperlink is described in a how-to; see [How to dynamically insert a hyperlink](#). This implies writing a script. For information about scripts, see "Writing your own scripts" on page 624.

Adding a personalized link

Personalized URLs (pURLs) are links that are tailor-made for a specific purpose, generally for individual clients. Typically, a pURL in a Connect template takes the user to a personalized landing page, for example, to download an invoice or get access to specific products or services. For more information, see "Personalized URL" on page 623.

Images

Images are a powerful ingredient in all of your templates. This topic explains how to add and use them. Currently the supported image formats are: **BMP**, **PNG**, **GIF**, **JPG/JPEG**, **TIF/TIFF**, **PDF**, **EPS** and **SVG**.

Ways to use images

In templates, both **imported** images and **external** images can be used (see "Adding images" on page 539 and "Resources" on page 318). Once added to the content of a template, an image can be resized (see "Resizing an image" on page 577) and alternate text can be linked to it (see "Setting an alternate text" on page 542).

Tip

Using images in an Email template? See "Using images in email campaigns: tips" on page 363.

Dynamic images

Images can be switched dynamically, so that a letter, email or web page can include one image or another, depending on a value in the data set. Read "Dynamic Images" on page 617 to find out how to add such switching images.

Background images

Several parts of templates, such as sections and media, and elements such as positioned boxes, can have a background image. Right-click the element and click the **Background** tab to select an image to be used as the element's background image. See "Background color and/or image" on page 579 and "Using a PDF file as background image" on page 339.

Tip

Editing PDF files in the Designer is not possible, but when they're used as a section's background, you can add text and other elements, such as a barcode, to them.

The quickest way to create a Print template based on a PDF file is to right-click the PDF file in the Windows Explorer and select **Enhance with Connect**. Alternatively, start creating a new Print template with a Wizard, using the PDF-based Print template (see "Creating a Print template with a Wizard" on page 327).

To use a PDF file as background image for an existing section, see "Using a PDF file as background image" on page 339.

Filling optional whitespace

Conditional content and dynamic tables, when used in a Print section, may or may not leave an empty space at the bottom of the last page. To fill that space, if there is any, an image or advert can be used as a 'whitespace element'; see "Whitespace elements: using optional space at the end of the last page" on page 344.

HTML tag: img

When you add elements, such as text, images or a table, to the content of a template, you are actually constructing an HTML file. It is possible to edit the source of the HTML file directly in the Designer; see "Editing HTML" on page 466.

In the section's source file, images are `` elements. The `` tag has at least four attributes: `src`, `alt`, `width` and `height`. `src` specifies the URL of the image. `alt` contains the alternate text; see "Setting an alternate text" on page 542.

The value of the attributes can be changed via a script; see "Attributes" on page 467.

Adding images

Imported or external images

In templates, both **imported** images and **external** images can be used.

Imported images are images that are saved within the template file. To import images into a template and add them to the content, you can use the drag-and-drop method or the Select Image dialog (both are explained below).

External images are either located on a specific website (URL), or in a folder on a hard drive that is accessible from your computer. Note that external images need to be available at the time the template is merged with a record set to generate output, and that their location should be accessible from the machine on which the template's output is produced. External images are updated (retrieved) when the output is generated. To refresh them at any other time, use the Refresh option in the menu (**View > Refresh**) or the Refresh button at the top of the Workspace. External images can not be added via the drag-and-drop method. Use the Select Image dialog instead (see below).

For information about referring to images in HTML or in a script, see "Resources" on page 318.

Via drag-and-drop

The drag-and-drop method is a quick way to import one or more images into a template.

1. Look up the image file or image files on your computer using the Windows Explorer.
2. Select the image (or images, using Shift+click or Ctrl+click) and drag the image file from the Explorer to the **Images** folder on the **Resources** pane at the top left.

3. To place an image in the content, drag it from the **Images** folder on the **Resources** pane to the content and drop it. The image will be inserted in the template at the position of the cursor.

Via the Select Image dialog

To either import an image into a template or use an external image in a template, the Select Image dialog can be used:

1. Position the cursor in the content where you want the image to be inserted.
2. On the **Insert** menu, click **Image**. Or, click the **Insert Image** button on the toolbar. The Select Image dialog appears.
3. Click **Resources**, **Disk** or **Url**, depending on where the image is located.
 - **Resources** lists the images that are present in the **Images** folder on the **Resources** pane.
 - **Disk** lists image files that reside in a folder on a hard drive that is accessible from your computer. Click the **Browse** button to select a folder (or an image in a folder). As an alternative it is possible to enter the path manually. The complete syntax is: file://<host>/<path>. Note: if the host is "localhost", it can be omitted, resulting in file:///<path>, for example: file:///c:/resources/images/image.jpg.
 - **Url** lists image files from a specific web address. Select the protocol (**http** or **https**), and then enter a web address (for example, <http://www.mysite.com/images/image.jpg>).
4. With an external image, you can check the option **Save with template**. If this option is checked, the file will be inserted in the **Images** folder on the **Resources** pane at the top left.

If not saved with the template, the image will remain external. Note that external images need to be available when the template is merged with a record set to generate output, and that their location should be accessible from the machine on which the template's output is produced. External images are updated (retrieved) at the time the output is generated.
5. Select the image from the list.
6. If the image is a PDF file that consists of more than one page, select the desired page.
7. Click **Finish**. The image will be inserted at the position of the cursor.

Using one file that contains a collection of images

When a template that contains lots of images is merged with a large record set, the many file requests may slow down the process of output generation. The solution is simple: combine the images into a single image file and display the part that holds the image. This reduces the number of file requests and can improve the output speed significantly.

For an explanation of how to do this, see "Optimizing a template" on page 954.

Moving an image

An image that is added to a section behaves like a character and is part of the text flow. To move it, simply click the image and drag and drop it somewhere else in the text flow. To learn how to wrap text around it, see "Wrapping text around an image" on page 577.

How to make an image stay at a certain position (like any image added to a Master Page) is explained here: "Pulling an image out of the text flow" on page 578. When an image has an 'absolute position' it can be moved around freely: hover over the border of the image to get a move pointer, click that pointer and drag and drop the image somewhere else.

Styling an image

Images can be resized using the handles on the sides and corners, or via the Image Formatting dialog, which opens when you right-click the image and select **Image...**, or select the **Format > Image** menu item.

Images can be styled using the same dialog, or through the CSS files; see "Styling templates with CSS files" on page 553.

A number of issues related to image styling are discussed in a separate topic: "Styling an image" on page 576.

Just like many other elements, images can be given borders and rounded corners, they can have inner and outer margins and they can be rotated. How to do this is described in general formatting topics, such as "Border" on page 580 and "Spacing" on page 591. All general formatting topics are listed under "Styling and formatting" on page 551.

Note

It is recommended to resize images outside of the Designer, with image editing software.

If necessary, it is possible to resize images automatically via a script in a Workflow process, as explained in a how-to: [How to resize images via a script](#).

Setting an alternate text

Once an image has been inserted in the content of a template, it can have an alternate text. The alternate text will be shown in emails and on web pages at the position of the image while the image is loading and when the image is not found. On web pages, alternate texts are also used for accessibility.

To set an alternative text, click the image and enter the alternate text in the **Alternate text** field on the **Attributes** pane at the top right.

Using a CSS gradient to create an image

CSS gradients are a new type of image added in the CSS3 Image Module. CSS gradients let you display smooth transitions between two or more specified colors, while repeating gradients let you display patterns. This way, using image files for these effects can be avoided, thereby reducing download time and bandwidth usage. In addition, objects with gradients look better when zoomed in a browser, and you can adjust your layout with much more flexibility.

For more information about the various types of CSS gradients and how to use them, see https://developer.mozilla.org/docs/Web/CSS/CSS_Images/Using_CSS_gradients.

Note

When CSS repeating gradients are displayed in a PDF reader, artifacts, like very thin lines may occur. When this happens, try setting the gradient's position a little bit different.

Table

Tables serve two different purposes: they are a way to display data in a tabular format, and they are also a way to position elements on a page.

In HTML email, Tables are the most reliable way to position text and images; see "Designing an Email template" on page 361. In web pages, on the other hand, Inline Boxes are the preferred way to position elements. Tables should only be used to display data in a tabular

format, not to position text and images. Tables used in web pages to position elements make those pages less accessible to users with disabilities and to viewers using smaller devices. In print, Tables can be used for both purposes.

There are two types of tables: standard tables which are static in nature, and Dynamic Tables which have a variable number of rows depending on a detail table in the record; see "Dynamic table" on page 618.

HTML element: table

When you add elements, such as text, images or a table, to the content of a template, you are actually constructing an HTML file. It is possible to edit the source of the HTML file directly in the Designer; see "Editing HTML" on page 466.

The HTML tag of a Table is `<table>`. Tables are divided into table rows with the `<tr>` tag. Table rows are divided into table data with the `<td>` tag. A table row can also be divided into table headings with the `<th>` tag.

The tags `<thead>`, `<tbody>` and `<tfoot>` can be used to group the header, body, or footer content in a table, respectively.

For information about HTML tables and a list of attributes, see http://www.w3schools.com/html/html_tables.asp.

Inserting a Table

1. On the toolbar, click the **Insert table** button, or on the menu select **Insert > Table > Standard**.
2. Enter the table's desired attributes:
 - **ID**: a unique identifier for the table. IDs are used to access the Table from scripts and as CSS selectors for style rules.
 - **Class**: A class identifier for the table. Classes can be shared between elements and are used to access the table from scripts and as CSS selectors for style rules.
 - The number of **rows** for the header, body and footer of the table.
 - The number of **columns**
 - The **width** of the table.

3. Check the option **Absolute** to give the Table an absolute position or use the **Location** drop-down to select where to insert the table:
 - **At cursor position** inserts it where the cursor is located in the template.
 - **Before element** inserts it before the HTML element in which the cursor is currently located. For example if the cursor is within a paragraph, the insertion point will be before the <p> tag.*
 - **After start tag** inserts it within the current HTML element, at the beginning, just after the start tag.*
 - **Before end tag** inserts it within the current HTML element, at the end, just before the end tag.*
 - **After element** inserts it after the element in which the cursor is currently located. For example if the cursor is within a paragraph, the insertion point will be after the end tag of the paragraph (</p>).*

* If the current element is located inside another element, use the **Elements** drop-down to select which element is used for the insertion location. The list displays every element in the breadcrumbs, from the current selection point until the root of the body.

Note

Tables on a Master Page have to have an absolute position, unless they are located inside another element with an absolute position.

4. Click **Next** and select which fields should show up in the Dynamic Table.
The order of the fields indicates in which order columns are displayed in the dynamic table, from left to right. Select a line and then use the **Up** and **Down** buttons to change the order of the columns.
You could change the placeholder for each data field as well; just click a placeholder to edit it.
5. Click **Next** and use the drop-down to select the desired table style.
6. Uncheck the box **Resizable columns** if the columns should not be resizable from the Design and Preview modes in the workspace. This is useful if the column size is determined in the Source mode or in a style sheet.
7. Click **Finish** to add the table to the section.

Header and footer

Adding a header or footer

To add a header or footer to an existing table, right-click the table and then select **Table > Insert thead** or **Insert tfoot**, on the shortcut menu.

Alternatively, click in one of the cells and select **Insert > Table > Insert thead** or **Insert tfoot**, on the menu.

Deleting a header or footer

To delete a header or footer, simply right-click the header or footer and select **Row > Delete** on the shortcut menu.

If the deleted element was targeted by a script, you will be asked if you want to delete the script as well.

Rows and columns

Adding a row or column

To add a row or column to an existing table, click in a cell. Then click the black triangle next to the **Insert Row Above** button on the toolbar, and click one of the **Insert** buttons, or select one of the options in the **Insert > Table Elements** menu.

Alternatively, right-click the table and on the shortcut menu, select **Row > Insert Above** or **Insert Below**, or select **Column > Insert Before** or **Insert After**.

Deleting a row or column

To delete a row or column, simply right-click the row or column and select **Row > Delete** or **Column > Delete** on the shortcut menu. If the deleted row was targeted by a script, you will be asked if you want to delete the script as well.

Styling a Table

To learn how to style tables, see "Styling a table" on page 571.

Resizing and moving a Table

Before you can resize or move a Table:

- Make sure that the position of the Table is absolute. If it's not, right-click the Table and on the shortcut menu, select **Convert to absolute**. (This option isn't available for Tables on a Master Page, as they must always have an absolute position, or be located inside another element with an absolute position.)
- Select the Table (see "Selecting an element" on page 469) and then, on the **Attributes** pane, check the option **Allow resizing**.

Resizing a Table

- Click in the table and drag the handles to resize it. Press the **Shift** key while dragging, to scale the table proportionally.
- Select the Table (see "Selecting an element" on page 469) and type the desired width and height in the respective fields on the **Attributes** pane.
- Select the Table and select **Format > Table**, on the menu. On the **Table** tab, change the **width** and **height** of the Table.

Moving a Table

- Click in the table and then drag the border to move the Table.
- Select the Table (see "Selecting an element" on page 469) and type the desired Y-offset and X-offset in the respective fields on the **Attributes** pane.
- Select the Table and select **Format > Table**, on the menu. On the **Table** tab, change the **Y-offset** and **X-offset** of the image.

Hiding the border

When using a Table to position other elements, you will want to hide the borders of the table. To do this, set the width of the border to 0; see "Border" on page 580.

Text and special characters

The vast majority of templates for personalized customer communications contain, of course, text. While the most common text element is a `<p>` or paragraph, other elements such as Headings (`<h1>` through `<h6>`) are also considered text elements. Text elements can be present within other types of elements such as table cells (`<td>`), boxes (`<div>`), etc.

Adding text

To add text, simply type in the workspace in the middle.

- Press **Enter** to insert a new paragraph.
- Press **Shift+Enter** to insert a line break.

Alternatively, copy-paste text into a template, or use the **Insert Lorem Ipsum** toolbar button to insert dummy text.

Text that precedes or follows the value of a **data field** can be added by the Text Script Wizard; see "Using the Text Script Wizard" on page 607.

Note: it is not possible to open a Word file in the Designer. When you copy text from a Word document, however, basic style characteristics travel with the content to PlanetPress ConnectDesigner. Formatting options like bold, italic and formats like Heading 1, Heading 2 are maintained.

Extra spaces

In HTML, extra spaces are generally removed. In Designer templates this is the same, because they are HTML files. In some cases however, you want extra spaces to be shown in your output. Read this how-to to learn how to maintain extra spaces in the text: [Maintain extra spaces in text](#).

Adding special characters

To add special characters:

1. Position the cursor where the character should be inserted.
2. On the **Insert** menu, point to **Special Characters** click **Symbols, Dashes and Spaces**, **Arrows**, or **Geometric Shapes**, and click one of the available special characters.

Adding page numbers

Page numbers can only be used in the Print context. See "Page numbers " on page 345.

HTML element: p, h, li and others

When adding elements, such as text, images or a table, to the content of a template, you are actually constructing an HTML file; see "Editing HTML" on page 466.

In HTML text can be contained in many different elements: paragraphs, span elements, line items and table cells, for example.

The HTML tag of a paragraph is `<p>`. The paragraph should be followed by a closing tag: `</p>`.

A line break looks like this in HTML: `
`.

Formatting text

Text can be styled, colored, centered, indented etc. It can even be displayed so that it reads from right to left. See "Styling text and paragraphs" on page 562.

In all templates you can use the fonts that are provided with the Designer, as well as imported fonts; see "Fonts" on page 587.

Snippets

A snippet is a small, ready-to-use piece of content in a file. Snippets can be re-used within the same template, in all contexts and sections. They can contain any contents that a section can have, such as text, images, variable data, dynamic tables, etc.

Normally, a snippet is an HTML file, but it can also be a JSON file.

When a snippet is added to different sections or contexts, it is displayed according to the section's or context's style sheet. This means that the same content can look different depending on the styles applied to the section or context, without changing the content.

Tip

It is possible to open and edit any HTML or JSON file in the Designer: select **File > Open**, select **All files (*.*)** as the file type and then select a HTML or JSON file.

Adding a snippet to the Resources

Before adding a snippet:

- Import the resource files that are related to the snippet, such as image files and CSS files, into the template file. Drag and drop the files to the corresponding folders (**Images** and **Stylesheets**, respectively) on the **Resources** pane. If you want to use external images, see "Images" on page 537.
- Drag the snippet itself to the **Snippets** folder on the **Resources** pane, or create a new snippet from an existing piece of content in the template (see "Creating a snippet" on the facing page).

Remote snippets

A remote snippet is an HTML file that is not located within your template file but is hosted on a Content Management System or other location.

To add a remote snippet:

1. Right-click the **Snippets** folder on the **Resources** pane, and click **New Remote Snippet**.
2. Enter a name for the file as it appears in the Snippets folder. This name is shown in the Snippets folder with the **.rhtml** file extension.
3. Enter the URL for the remote resource. This must be a full URL, including the **http://** or **https://** prefix, domain name, path and file name.

Note

Remote snippets may contain other resources, such as images. There is one limitation though: only **absolute** paths are supported inside remote snippets. Images and other resources referred to with a relative path (for example: `images/img.gif`) or root-relative path (any path starting with a slash, for example: `/base/images/img.gif`) won't be available in the template.

Adding a snippet to a section

Drag-and-drop

To add the snippet to the content of a section, drag the snippet from the **Snippets** folder on the **Resources** pane to the desired location in a section.

Check the option **Insert as shared content** to insert a **reference** to the original snippet in the template, rather than a **copy** of the original snippet.

When a snippet is being used as shared content, the contents of the snippet itself are not added to the page. Modifying the snippet on the page actually modifies the snippet's source. If a snippet is used in multiple locations (such as different contexts and sections), modifying one instance will modify all of them at once.

Note

Remote snippets inserted as shared content cannot be changed from within the Designer. Of course, their source file can be edited outside of the Designer. When that happens, the changes will become visible in remote snippets that are inserted as shared content.

Via a script

In addition to the drag-and-drop method, it is possible, and often useful, to insert a snippet or part of it, using a **script**; for remote snippets this is normal practice. See "Loading a snippet via a script" on page 640.

Tip

To **export** a snippet from your template, drag or copy/paste it out of the Snippets folder to a folder on the local hard drive.

Creating a snippet

To turn a parts of a letter, email or web page into a snippet for reuse in the content of a template:

1. Select the part that should be saved in a snippet.
2. Right-click the selection, point to **Snippet** and click **Create**.

3. Right-click the new snippet on the **Resources** pane in the **Snippets** folder and rename it.

JSON Snippets

JSON Snippets are snippets that contain pieces of JSON data instead of HTML. Just like HTML snippets, JSON snippets are stored in the **Snippets** folder on the **Resources** pane, but their file name should end in '.json'.

JSON Snippets cannot be inserted into the content directly, but they can be accessed via a script using the function **loadjson()**:

```
var json_data = loadjson("snippets/snippet.json");
results.html(json_data.field1);
```

See also: "Writing your own scripts" on page 624.

For an example in which JSON snippets are being used to localize a template, see this how-to: [Localizing templates using json](#).

Styling and formatting

In the Designer you have everything at hand to make your templates look good: colors, fonts and all the tools to position, align and embellish elements in your designs. This topic informs about the ways to style a template.

Local formatting versus style sheets

There are in general two ways to style elements:

- Using **local formatting**. Local formatting means styling an element directly, using a toolbar button or one of the formatting dialogs.
- Using **Cascading Style Sheets (CSS)**. Style sheets can determine the appearance of individual elements, as well as the appearance of elements that have the same class or HTML tag.

Whether applied through style sheets or through local formatting, behind the scenes all layout properties in the Designer are CSS properties. When you format an element locally, an **inline** style rule is added to the element.

Note that where local formatting conflicts with a formatting rule for the same element in one of the style sheets, the local formatting rule gets priority; the rule in the style sheet will be ignored.

It is highly recommended to use style sheets in templates right from the start. Even more so if the communications are going to be output to different output channels, or if they consist of different sections (for example, a covering letter followed by a policy). Using CSS with templates allows a consistent look and feel to be applied. A style sheet can change the look of multiple elements, making it unnecessary to format each and every element in the template, time and again, when the company's layout preferences change. See "Styling templates with CSS files" on the next page.

Layout properties

Colors and fonts make an important contribution to the look and feel of your template. See "Colors" on page 583 and "Fonts" on page 587.

Text and paragraphs have a number of formatting options that are not available for other elements: font styles and line height, for example. See "Styling text and paragraphs" on page 562.

Boxes and a number of other elements can have a background color and/or background image; see "Background color and/or image" on page 579.

Several elements, such as boxes, images, paragraphs, and tables, can have a border; see "Border" on page 580.

Boxes, images, tables, text and other elements can be rotated; see "Rotating elements" on page 570.

Spacing (padding and margin) helps to position elements relative to other elements in the template; see "Spacing" on page 591.

The best way to position elements depends on the output channel for which the template is intended; see "How to position elements" on page 567.

The locale setting influences how dates, numbers and amounts of money are displayed; see "Locale" on page 590.

Styling templates with CSS files

The Layout toolbar and the Format menu offer many possibilities to style every piece of a template. However, styling every single element, one after another, is a lot of work and, more importantly, can result in a template with a messy mix of styles that isn't easy to maintain and lacks consistent design. Therefore the preferred way to style templates is with CSS files: Cascading Style Sheets. This topic explains how to do that.

Why use CSS files

The basic idea behind CSS is to separate the structure and contents of a (HTML) document as much as possible from the presentation of that document.

Cascading Style Sheets were originally designed for use with web pages, or HTML files. Since every template in the Designer is constructed in HTML, CSS files can also be used in the Designer.

Instead of setting the font size, line height, color etc. for each and every paragraph in the template itself, you can define a layout for all paragraphs, and for all output channels, in a CSS file.

The benefit of this is that you can quickly and easily change the look and feel of all contexts in one template, without having to change the contents. In the event that your company chooses to use another font or to adjust its corporate colors, you only have to change the style sheets.

You are writing HTML

When you add elements, such as text, images or a table, to the content of a template, you are actually constructing an HTML file.

To see this, toggle to the **Design** tab in the workspace. Click anywhere in the content. Take a look at the *breadcrumbs* at the top of the workspace. The breadcrumbs show the HTML tag of the clicked element, as well as the HTML tags of other elements to which the clicked element belongs. The clicked element is at the end of the line.

To edit the HTML text directly:

- In the workspace, toggle to the **Source** tab.

On this tab you can view and edit the content of the template in the form of plain text with HTML tags (note the angle brackets: <>). You may add and edit the text and the HTML tags, classes, ID's and other attributes.

To learn more about HTML, see for example <https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Introduction> and <http://www.w3schools.com/html/default.asp>.

Many video courses and hands-on courses about HTML (and CSS) are offered on the Internet as well, some for free. Go, for example, to www.codeschool.com or www.codecademy.com and look for HTML (and CSS) courses.

What you can't do with CSS

In Connect, it depends on the output channel what can and cannot be done with CSS. CSS can only be used to its full potential with HTML output. Animation and transition features won't work in Print output, obviously.

Included Cascading Style Sheets

When you create a template, a number of style sheets is automatically included:

- One style sheet that applies to all document types: `context_all_styles.css`.
- One or more style sheets specific to the context (Print, Email, Web). For example, when you create an action email using the Wizard, the files `context_htmlmail_styles.ccs` and `basic_email_action.css` are automatically added to the **Stylesheets** folder on the **Resources** pane.
- A style sheet that defines default styles for tables: `default.css`. It contains the styles that you can choose from when you insert a table via the **Insert** menu or the **Insert table** toolbar button.

Note

Do not change the `default.css` style sheet. Use the global style sheet or the style sheet for the relevant context to define your own styles for tables.

Adding CSS files

To add a CSS file of your own, open an Explorer window, **drag** the file to the **Resources** pane and drop it on the **Stylesheets** folder.

To create a new CSS file, right-click the **Stylesheet** folder on the **Resources** pane and select **New Stylesheet**.

Note

The order in which style sheets are executed, can affect the actual output. This sequence can be set per section; see "Applying a style sheet to a section" on page 323.

Tip

- To export a CSS file from your template, drag or copy/paste it out of the Stylesheets folder to a folder on the local hard drive.
- It is possible to open and edit any CSS file in the Designer: select **File > Open**, select **All files (*.*)** as the file type and then select a CSS file.

Using a remote style sheet

A remote style sheet is not located within your template but is rather hosted on an external web server (generally called a **CDN**). When generating Web output, these files are referenced in the web page's header and are served by the remote server, not by the PlanetPress Connect Server module.

To add a remote style sheet:

1. Right-click the **Stylesheet** folder on the **Resources** pane, and click **New Remote Stylesheet**.
2. Enter a name for the file as it appears in the Stylesheet resources. For better management, it's best to use the same filename as the remote resource.
3. Enter the **URL** for the remote resource. This must be a full URL, including the http:// or https:// prefix, domain name, path and filename.

4. Optionally, for a Capture OnTheGo Form, you can check **Use cached Capture OnTheGo resource**, to prevent downloading a remote style sheet again if it has been downloaded before. The file should be available on a publicly accessible location, for example: a folder location on a corporate website, hosted by a CDN (Content Delivery Network) or shared via a Workflow process.

There are a few advantages to remote resources:

- These resources are not served by your server, saving on space, bandwidth and processing.
- Using a popular CDN takes advantage of caching - a client having visited another website using that same CDN will have the file in cache and not re-download it making for faster load times for the client.

Tip

To refresh the remote resources in a Designer view, use the Refresh option in the menu (**View > Refresh**) or the Refresh button at the top of the Workspace.

Styling your templates with CSS files

Note

Email clients do not read CSS files and some even remove a <style> tag when it is present in the email's header. Nevertheless, CSS files can be used with the Email context in the Designer. When generating output from the Email context, the Designer converts all CSS rules that apply to the content of the email to inline style tags, as if local formatting was applied.

Step 1: edit CSS

Editing CSS using a property sheet

1. Select **Edit > Stylesheets**.
2. Click the downward pointing arrow next to **Global** and select the context that you want to edit styles for, or select the Global CSS file to edit CSS rules that apply to all contexts.
3. Click **New**, or click one of the selectors that are already listed and click **Edit**.

4. Type a CSS selector. This can be:

- A class: `.class`. Class rules apply to all HTML elements with that class. When you create a class, choose a name that indicates what the class is used for, e.g. 'small' for a class that gives elements the font size 'small'. The class name has to be preceded by a dot, e.g. `.small`.
- An ID: `#id`. An ID is always preceded by `#`, e.g. `#sender`. When you create an ID, choose a name that indicates what the ID is used for, e.g. `#sender` would refer to the HTML element with information about the sender.

Note

Each ID should be unique and can only be used once in each section.

- An HTML element: `p`, `h1`, `table`, etc. Type the tag name without the angle brackets.
 - A combination of HTML elements, separated by a comma. The CSS rule will apply to all HTML elements that are listed in the selector. For instance, a CSS rule with the selector `h1, p` applies to first level headings as well as paragraphs.
 - HTML elements inside other HTML elements. For instance, a rule for all paragraphs inside a `div` element has the selector: `div p`.
 - Etcetera. See http://www.w3schools.com/cssref/css_selectors.asp for more CSS selectors and combinations of CSS selectors.
5. Select the layout options that should apply to selected elements; see "Styling and formatting" on page 551. Note: where a width can be set as a percentage, it is a percentage of the space between the left and right margin.
6. Click **OK**.
7. In the Stylesheets dialog, click the selector that you chose. All CSS rules for that selector will become visible in a box below the list of selectors.

Editing plain CSS

- Click the button **Advanced** in any property sheet to open a CSS property editor. Type CSS properties at the left and values at the right.
- In the **Resources** pane at the left, double-click the global stylesheet or the stylesheet for the relevant context. The file opens in the workspace in the middle.

A list of all CSS properties and their possible values can be found here:
<http://www.w3schools.com/cssref/>.

Step 2: apply CSS to the content

After editing the CSS file(s), make sure that the CSS rules actually apply to one or more elements in the template.

- CSS rules for HTML elements, such as paragraphs, are automatically applied to all elements with the corresponding HTML tag.
- To make a CSS rule for a certain class or ID work for an element in your document, you have to add the class or ID to that HTML element (as described below).

Note

Classes may be reused throughout one section, but a specific ID should not be used more than once in each section. CSS layout rules for an element with a certain ID only apply to the first element with that ID in each section. If you have two sections inside of a Print context, then you can have the same ID on two sections; they will both be affected by the CSS rules for the element with that ID.

- Style sheets only apply to sections in which they are included; see "Applying a style sheet to a section" on the next page.

Adding a class or ID to an HTML element

1. Select the element (see "Selecting an element" on page 469).
2. On the **Attributes** pane, type the **ID** and/or **class**. Type the ID **without** the preceding # and class names **without** a dot.

Note

Note: Elements can have multiple classes. Separate the class names with a space (eg. "red small").

Alternatively, after selecting an element, you can click the **Source** tab at the bottom of the workspace. The selected element will be highlighted in the source. Add the class or classes and/or the ID to the opening tag of the HTML element, for example: `<p class="intro">`.

Applying a style sheet to a section

In order for a style sheet to be applied to a specific section, it needs to be included in that section. There are two ways to do this.

Drag & drop a style sheet

1. Click and hold the mouse button on the style sheet on the **Resources** pane.
2. Move the mouse cursor within the **Resources** pane to the section to which the style sheet should be applied.
3. Release the mouse button.

Using the Includes dialog

1. On the Resources pane, right-click the section, then click **Includes**.
2. From the **File types** dropdown, select **Stylesheets**.
3. Choose which CSS files should be applied to this section. The available files are listed at the left. Use the arrow buttons to move the files that should be included to the list at the right.
4. You can also change the order in which the CSS files are read: click one of the included CSS files and use the **Up** and **Down** buttons. Note that moving a style sheet up in the list gives it **less** weight. In case of conflicting rules, style sheets read later will override previous ones.

Note

Style sheets are applied in the order in which they are included in a section. The styles in each following style sheet add up to the styles found in previously read style sheets. When style sheets have a conflicting rule for the same element, class or ID, the **last** style sheet ‘wins’ and overrides the rule found in the previous style sheet.

Note

Which style sheets are included can also be set for the **Web context** as a whole: in step 1, right-click the Web context, instead of a section.

How to determine which styles are applied

To see which styles are applied to an element, select the element (see "Selecting an element" on page 469) and take a look at the Styles pane that sits next to the Attributes pane.

The Styles pane shows which CSS style rules apply to the currently selected element. A link next to a style rule will open the file where that particular style is defined. This can be either a CSS file or the source file of a section if local formatting was used (see "Styling and formatting" on page 551).

A crossed-out style rule signals that it was overruled by another style rule. This happens when:

- A more specific, and therefore more important rule, is encountered for the same element. See "Using a more specific CSS rule" below to learn more about the specificity of style rules.
- A rule with the same importance is read after the first rule. Not only is the order of the rules in a CSS file important, but also the order in which the style sheets are read. The style sheets that are included with a section are read in the specified order; see "Applying a style sheet to a section" on page 323.

Using a more specific CSS rule

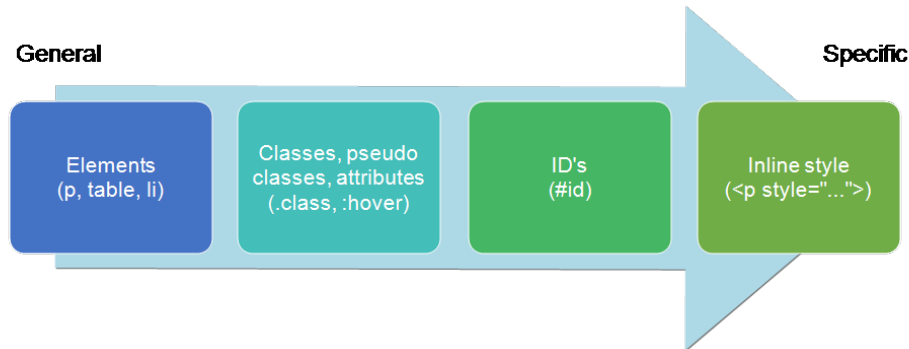
By default, many CSS properties of an HTML element also apply to the elements inside that element. For example, a CSS rule that specifies a certain font-type for a box also applies to paragraphs in that box. In this example the box is the 'parent' element and the paragraphs are the 'child' elements that inherit the font-type property of the box.

Note

Although the background color property seems to be inherited, it isn't. Most elements are transparent; therefore the background color of the parent element shines through.

To replace inherited style properties, you need to add a more specific CSS rule for that (type of) element. In case of a conflict between a general rule and a more specific rule, the more specific rule will be applied.

The following diagram shows the order of specificity.



Rules for HTML elements (p, table, li etc.) are general rules. Rules for classes, pseudo classes, and elements with a certain attribute (.class, :hover, [target]) are more specific. Rules for elements with a certain ID are even more specific. The most specific are inline styles.

Example

Assuming that a table has the CSS property "color: red" (which colors text in the cells red), a more specific rule for cells in that table could be, for example:

- A rule for the text color of all table cells (td elements), for example: `td { color: green; }`.
- A rule for the text color of table cells with a certain class, for example `.green { color: green; }`.
- A rule for the text color of a table cell with a certain ID, for example: `#greentext { color: green; }`.
- An inline style rule (local formatting) added to the HTML tag of a particular table cell, for example: `<td style="color: green;">...</td>`.

Each of these rules is more specific than the previous rules. All of these rules are more specific than the rule that applies to the table as a whole.

Note

When `!important` is added to a style rule (e.g. `color: red !important;`), this rule overrides any other style rules, even inline style rules.

Styling text and paragraphs

There are numerous ways to format text in a template. You can apply a certain font, make text bold, transform it to uppercase, center it, color it, etc.

This topic explains how to apply local formatting to text. It is recommended though, to format text using style sheets; see "Styling and formatting" on page 551 and "Styling templates with CSS files" on page 553.

Tip

With the **Copy fit** feature, text can automatically be scaled to the available space in a Box or Div. See "Copy Fit" on page 566.

Formatting text and paragraphs locally

An intuitive way of formatting text locally is by using the toolbar buttons: select some text, or an element that contains text (see: "Selecting an element" on page 469) and click one of the toolbar buttons to make it bold, center it, create a numbered or bulleted list, etc.

To quickly change a paragraph into a Heading, Address or Pre element, select the paragraph (see: "Selecting an element" on page 469) and on the **Format** menu, select the appropriate element.

More local formatting options are available in the Formatting dialogs; see below.

Formatting text

To open the Text Formatting dialog, select some text and then select **Format > Text**. In the Text Formatting dialog you can set:

- The font, font size, color and background color:
 - **Font:** Select the font used to display text. See also: "Fonts" on page 587. This is equivalent to setting the `font-family` property in CSS.
 - **Font size.** Enter the size in a measure, named size or percentage. This is equivalent to setting the `font-size` property in CSS.
 - **Color:** This the color of the text. Select a named font color as defined in the Edit Colors dialog (see "Colors" on page 583) or click the colored square to create a new color or to enter a color manually. The color value must be a valid HTML color name or hexadecimal color code. This setting is equivalent to the `color` property in CSS.
 - **Background color:** This is the background color of the text. Select a named font color as defined in the Edit Colors dialog (see "Colors" on page 583) or click the colored square to create a new color or to manually enter a color value (a valid HTML color name or hexadecimal color code). This setting is equivalent to the `background-color` property in CSS.
- The spacing between letters and words and the way the text is wrapped:
 - **Letter Spacing:** The space between characters in a text in measure or percentage. This is equivalent to the `letter-spacing` property in CSS.
 - **Word Spacing:** Set the space between each word in a text in measure or percentage. This is equivalent to the `word-spacing` property in CSS.
 - **Whitespace:** Specify how the text wraps. See [CSS White-Space](#) for details. This is equivalent to the `white-space` property in CSS.
- The style of the text. Check any option to apply the selected style to text within the element. This list shows the CSS property and value for each of the options:
 - **Bold:** Sets the `font-weight` to `700`.
 - **Italic:** Sets the `font-style` to `italic`.
 - **Underline:** Sets the `text-decoration` to `underline`.
 - **Strikethrough:** Sets the `text-decoration` to `line-through`.
 - **Subscript:** Sets the `vertical-align` to `super`.
 - **Superscript:** Sets the `vertical-align` to `sub`.
 - **Capitalize:** Sets the `text-transform` to `capitalize`.
 - **Uppercase:** Sets the `text-transform` to `uppercase`.

- **Lowercase:** Sets the `text-transform` to lowercase.
- **Small-caps:** Sets the `font-variant` to small-caps.

Note

All settings in the Text Formatting dialog are in fact CSS style rules. When you change one or more settings, the selected text gets wrapped in a Span element that has an **inline** style tag containing the selected setting(s). Click the **Advanced** button to add CSS properties and values to the inline style tag of the Span directly. For more information about CSS, see "Styling and formatting" on page 551.

Formatting a paragraph

Through the Paragraph Formatting dialog you can set the line height and first indent, among other things. It also lets you add spacing and a border; see "Spacing" on page 591 and "Border" on page 580.

To open the Paragraph Formatting dialog, select a paragraph (see: "Selecting an element" on page 469) or place the cursor in a paragraph, and then select **Format > Text**.

On the **Formats** tab you can set:

- **Line-height:** Specify the height of each line in the paragraph's text, in a measure or percentage. Note that this is not the spacing between lines, but rather the complete height of the line itself including the text. This is equivalent to the `line-height` property in CSS.
- **Align:** Select how text should be aligned, such as left, center, right or justify. Equivalent to the `align` property in CSS.
- **First Indent:** Specify the indentation of the first line of the paragraph. Equivalent to the `text-indent` property in CSS.
- **Display:** Select how to display the element. This can also be used to hide an element completely using the none option. See CSS Display. Equivalent to the `display` property in CSS.
- **Direction:** Select in which direction text should be displayed (left to right, right to left, or auto). Useful for certain languages such as Arabic, Hebrew, etc. This is equivalent to the `dir` HTML attribute.

- **(Page) breaks:** these settings are only useful in Print sections, as only Print sections have pages.
 - **Before:** Sets whether a page break should occur **before** the paragraph. This is equivalent to the `page-break-before` property in CSS; see [CSS page-break-before property](#) for an explanation of the available options.
 - **Inside:** Sets whether a page break is allowed inside the paragraph. Equivalent to the `page-break-inside` property in CSS; see [CSS page-break-inside property](#) for an explanation of the available options.
 - **After:** Sets whether a page break should occur **after** the paragraph. Equivalent to the `page-break-after` property in CSS; see [CSS page-break-after property](#) for an explanation of the available options.
 - **Widows and orphans:** Keeps lines of text together; see "Preventing widows and orphans" on page 347 for an explanation.

Note

For more information on page breaks, widows and orphans, see the [W3 Paged Media reference](#).

Click the **Advanced** button to add CSS properties and values to the inline style tag directly.

Removing local formatting from text

Layout buttons and options on the Format menu add **inline** style tags to the text. Style tags can look like this: `...` or like this: `<p style="color: red;" >`.

Inline style tags have priority over styles defined in a CSS file. For example, when a formatting rule in a style sheet colors all paragraphs green, a paragraph with an inline style tag to color it red would still stay red. So, when a rule in a style sheet doesn't seem to work, an inline style tag can be the culprit. In that case you might want to remove the local formatting.

To remove local formatting:

- Select the formatted text and click the toolbar button **Remove Formatting**. Doing this removes inline style tags from the selection.

- Alternatively, click the **Source** tab at the bottom of the workspace (or select **View > Source View**) to manually remove style tags.

Tip

When you select an element in the template, the **Styles** pane will show which styles are applied to that element. The link behind the style will take you to the place (the Source tab, or a CSS file) where that style is defined.

Copy Fit

Copy Fit is a feature to automatically scale text to the available space: the name of a person on a greeting card for example, or the name of a product on a shelf talker.

This feature is only available with Box and Div elements in Print sections.

Activating the Copy Fit feature

After adding a Box or Div element to a Print section (see "Boxes" on page 513), you can activate the Copy Fit feature on that element. Text inside that Box or Div or text in an element inside it, will then be scaled to fit the available space. This is how it's done:

1. Right-click the Box or Div element and click the respective element on the shortcut menu. Alternatively, select the element (see "Selecting an element" on page 469) and on the **Format** menu click the respective element.
2. Click the **Content** tab.
3. Check the **Copy Fit** option.
4. Enter the **Min font size and/or Max font size** using a valid font measurement unit (pt, px, in, cm or mm). Do not put a space between the number and the unit.
 - The **minimum font size** is 1pt. The default minimal font size is 4pt.
When the minimum font size is left **blank**, the font size in Design view becomes the minimum font size. This means that the text can only be made bigger than its initial size.
 - The **maximum font size** is 1048pt. By default this is 48pt.
When the maximum font size is left **blank**, the font size in Design view becomes the maximum font size, so that the text can only be made smaller than its initial size.
5. When the option **Fit to width only** is checked, no line breaks will be added to the text.

6. Optionally, you can specify a **child** (an element inside the Box or Div) by giving its ID, for example: **#product**, or class, for example: **.product** - note the dot. The Copy Fit feature will only be applied to this child element.

Tip

To give an element a class or ID, select it and add the class or ID on the Attributes pane. An ID is meant to be used once in each section, while a class can be shared between several elements.

7. Click **Apply** or **OK**. Note that the effect of the Copy Fit feature can only be seen in Preview mode.

If it is impossible to make a text fit within the box with the given minimum and maximum font size, this will be reported as an error during a preflight (see "Testing scripts" on page 632).

How it works

When the Copy Fit feature is activated, the font size is calculated for the entire Box. Elements inside that Box get a font size relative to the Box. This means that their relative proportions are maintained.

How to position elements

To position elements in relation to each other in a template, wrap those elements in a Table or Box (see "Table" on page 542 and "Boxes" on page 513) and/or use the Spacing property of the elements. The Spacing property can also be used to indent elements or create a hanging paragraph or image; see "Spacing" on page 591. Guides help to align elements as well; see below.

Where to use Tables and Boxes

Tables, Positioned Boxes and Inline Boxes can help position elements in relation to other elements. It depends on the context which element is best to use.

In the Email context, Tables are the most reliable way to position text and images; see "Designing an Email template" on page 361 and "Table" on page 542.

In the Web context, Inline Boxes are the preferred way to position elements; see "Boxes" on page 513. Tables should only be used to display data in a tabular format, not to position text and images. Tables used in web pages to position elements (and often, Positioned Boxes) make those pages less accessible to users with disabilities and to viewers using smaller devices.

In the Print context, Tables can be used to position elements, as well as both types of Boxes; see "Table" on page 542 and "Boxes" on page 513.

Spacing

Boxes, tables, paragraphs and many other elements have a **margin** and **padding**.

The margin is the white space around an element, outside the border. It is used to position an element in relation to the other elements, by putting more space between the element and its surrounding elements.

The padding is the space between an element's content and its border. It is used to position the content of the element inside the border.

To learn how to set an element's spacing properties, see "Spacing" on page 591.

Tip

Use a negative left margin to create a hanging paragraph or image.

Guides

Guides are horizontal and vertical lines used to help in designing templates, for example when positioning absolute positions boxes over a PDF background. They can only be used in Print sections.

- Select **View > Guides > Show guides** to show or hide the guides and margins.

To **add** a guide, press the **Insert Horizontal Guide** or **Insert Vertical Guide** buttons on the Toolbar.

To **move** a guide, click and drag it to a new location.

Click the **Shift** key while dragging to make the guide snap to the closest ruler tick.

Double-clicking a guide brings up its Edit dialog where its exact position can be adjusted.

- Select **View > Guides > Lock guides** to lock the guides in their current position.
- Select **View > Guides > Snap to guides** to make Positioned Boxes (and any other objects that have their **position** set to **absolute**) snap to guides when moved within a few pixels of them.

To **delete** a guide, double-click on it and press the **Delete** button.

Using the CSS `position` property

An element can be positioned independently of the text flow by changing its `position` property to `absolute` or to `relative` (to its parent).

When an element is placed inside another element, such as a Box, changing its `position` property to `absolute` positions the element absolutely inside its 'parent'.

With the `position` property of an element set to `absolute`, the `top` or `bottom` and `left` or `right` properties position the element inside its parent with exact values: pixels (px), centimeters (cm), etc. Negative values are allowed.

For an explanation of all values that the `position` property can possibly have, see http://www.w3schools.com/css/css_positioning.asp.

Where to use it

In Print sections, setting the `position` property to `absolute` can be very useful. It takes the element out of the text flow, so that the element stays where it is on the page. On Master Pages (which are only used in Print sections) elements are always positioned absolutely; if not, they must be located inside an element that has an absolute position.

In Web sections, setting the `position` property to `absolute` may sometimes be useful for elements inside a Div element, but in general, elements should not be positioned absolutely. Designs for the Web should be flexible so that they display nicely on a variety of devices and screen sizes.

In Email sections, do not use this property. Use Tables instead (see "Designing an Email template" on page 361 and "Table" on page 542).

How to use it

In the Formatting dialog the `position` property can often be found on the first tab, under **Positioning**. To open the Formatting dialog, right-click the element and click the respective element on the shortcut menu. Alternatively, select the element (see "Selecting an element" on page 469) and on the **Format** menu click the respective element.

This property isn't present in one of the tab menus of the style rule editor, but you can add it and specify a value after clicking the Advanced button in the style rule editor (see "Styling templates with CSS files" on page 553).

About the CSS `display` property

The `display` property is one of the most important CSS properties for controlling layout. Yet it is unlikely that you will use it often to position elements in a template: in most cases, the initial value of the `display` property for an element will be the right one.

It is more likely that you will use this property in style sheets and scripts to hide certain elements, by setting the value of this property to `none(display: none;)`. (See "Styling templates with CSS files" on page 553 and "Writing your own scripts" on page 624.)

For an online tutorial about this property, see [w3schools website](#).

Rotating elements

In any type of template, boxes, images, tables, text and other elements can be rotated.

The toolbar buttons **Rotate Clockwise** and **Rotate Counter Clockwise** rotate the element in which the cursor is located 90 degrees at a time.

To rotate an element into another angle position, use the 'angle' CSS property of the element. In most cases, this can be done in the element's Formatting dialog. In other cases, such as with text, you have to enter the CSS property and value manually. Both methods are explained in the following procedure.

1. Right-click the element and click the respective element on the shortcut menu. Alternatively, select the element (see "Selecting an element" on page 469) and click the respective element on the **Format** menu.
2. On the first tab, look for the **angle** property. If it is available, type the number of degrees the element should be rotated. A positive number will rotate the element clockwise, a negative number rotates it counter-clockwise. Skip steps 3 to 6. If the angle property is not available, proceed with the following step.

3. Click the **Advanced** button to open the Advanced Formatting dialog.
4. Click in the first blank field under **Property** and type **transform**.
5. Click in the field next to it, under **Value** and type **rotate(**, followed by the number of degrees the element should rotate, and then **deg)**, for example: `rotate(20deg)`. A positive number will rotate the element clockwise, a negative number rotates it counter-clockwise.
6. Close the Advanced Formatting dialog.
7. Close the Formatting dialog, or click the Apply button to see the effect without closing the dialog.

Note

It is also possible to rotate elements by creating a style rule in a style sheet; see "Styling templates with CSS files" on page 553.

Styling a table

Just as other elements, tables can be styled in two ways:

- With **local formatting**. This means styling the table directly, using the Formatting dialog.
- Via **Cascading Style Sheets (CSS)**. In a style sheet, style rules are declared for elements with different HTML tags, ID's and classes.

These two methods are described below. See "Styling and formatting" on page 551 for background information about these two methods.

Selecting a table, row or cell

There are several ways to select a table or row:

- Click in the table or row. Then, in the **breadcrumbs** (see "Selecting an element" on page 469) click **table** to select the table, or **tr** to select the row.
- Right-click a cell and from the shortcut menu, choose **Table > Select** or **Row > Select**.
- Click in a cell and then use the toolbar: click the **Select Table** button or click the black triangle next to that button and then click **Select Table** or **Select Row**.

Selecting one cell is easy: just click in it.

Tip

Use the Styles pane to see which styles apply to the currently selected table, row or cell.

Via the Formatting dialog

The Formatting dialog allows you to change the font, font size and color (see "Fonts" on page 587), the borders (see "Border" on page 580), the cell padding (the distance between the edge of the cell and its content, see "Spacing" on page 591), and the background color or image of the table and its cells ("Background color and/or image" on page 579).

To open the Formatting dialog for **one cell** or for the **table as a whole**:

- Click in a cell and choose **Format > Table** or **Format > Table Cell**.
- Right-click it and choose **Cell...** or **Table...** from the shortcut menu.

Note that in this case **Table** styles the table as a whole. When you choose **Table** and change the border, for example, the borders of the cells inside it will not be changed.

To style **all cells** in a table or row at the same time via the Formatting dialog, you have to select the table or row first; see "Selecting a table, row or cell" on the previous page. Next, to open the Formatting dialog, choose **Format > Table Cell**. The settings that you make now will be applied to all cells in the selected row or table.

For information about specific options in the formatting dialogs, see "Table Formatting dialog" on page 732 and "Table Cell Formatting dialog" on page 735.

Via a style sheet

Cascading Style Sheets (CSS) offer more ways to style a table and its contents, than the Formatting dialog does. This is especially true for Dynamic Tables. With local formatting, all rows that are added on the fly (in Preview mode and in output) will look exactly the same as the first one. Alternating row colors, for example, in dynamically added rows can only be done via CSS. How to do this is described below.

Another good reason to prefer style sheets over local formatting for Dynamic Tables, is that the output from a Dynamic Table is created slightly faster when it's styled via Cascading Style Sheets than when it's styled with local formatting.

How to use style sheets is explained in another topic; see "Styling templates with CSS files" on page 553.

Note that to make a style rule apply to a **specific** table, row or cell, you have to add an ID or class to that table, row or cell.

Adding an ID or class to a table, row or cell

A style sheet contains a bunch of style rules for different elements, that are identified via a CSS **selector**. This can be the element's HTML tag (without the angle brackets), ID or class.

When used as a CSS selector, the HTML tag for a table is **table**. For a row, it is **tr** and for a cell, **td**. A style rule that uses one of these, however, would apply to **all** tables, rows, or cells. For a rule to be more specific you need to add an ID (for a unique element) or a class (for a set of similar elements) to the table, row or cell, and use that as the style rule's selector.

Before you can add an ID or class to a table, row or cell, you have to select that table, row or cell (see "Selecting a table, row or cell" on page 571). After selecting the cell, row or table, type the ID or class in the respective field on the **Attributes** pane.

In CSS, refer to the table, row or cell with `#ID` (where ID should be replaced with the actual ID) or with `.class` (where class should be replaced with the actual class).

Styling the first, last and nth rows

The CSS pseudo-classes `:first-child`, `:last-child` and `:nth-child()` are very useful for styling table rows, especially in Dynamic Tables.

A CSS **pseudo-class** follows a selector to specify a special state of that selector. It always starts with a colon.

The pseudo-classes `:first-child`, `:last-child` and `:nth-child()` select an element only if it is the first, last or nth child element respectively. (In HTML and CSS, the word **child** refers to an element inside another element.)

The following CSS style rule selects the table row (tr) that comes first (`:first-child`) in its parent (which naturally is a table), and colors its background red:

```
tr:first-child {
    background: red;
}
```

Tip

In a Dynamic Table, data are in the body of the table (selector: tbody) and subtotals are in the footer (selector:tfoot).

Selecting a specific row, odd or even rows, or every nth row

The pseudo-class `:nth-child()` lets you select a specific row, all odd or even rows, or every nth row.

Between the round brackets in `:nth-child()` you can fill in a number, `odd` or `even`, or a formula: a_n+b . In the formula, a represents a cycle size (every...), n is a counter (for the child elements), and b is an offset value ('start at b '). The following examples will make this clear.

`:nth-child(3)` matches just one element: the third child element.

`:nth-child(odd)` matches child elements 1, 3, 5, 7, etc. The keyword `odd` substitutes the expression $2n+1$, which in other words says: 'take every second element, starting at 1'.

`:nth-child(even)` matches child elements 2, 4, 6, 8, etc. The keyword `even` substitutes the expression $2n+0$, or simply $2n$.

`:nth-child(3n)` matches child elements 3, 6, 9, 12 etc.

`:nth-child(3n+1)` matches child elements 1, 4, 7, 10 etc., so every third element, starting at 1.

Via script (based on a data field value)

To style a table, row or cell based on a data field value, you have to write a script (see "Writing your own scripts" on page 624).

First add an ID or class to the table, row or cell that needs to be styled: select the element (see "Selecting a table, row or cell" on page 571) and add an ID on the **Attributes** pane. Then create a script, using that ID or class as the script's selector. The script can be very simple:

```

if (record.fields.COUNTRY == 'CANADA') {
    results.css('color', 'green');
}

```

The Designer Scripts API provides several functions to style elements, for example `css()`, `hasClass()` and `addClass()` (see "Designer Script API" on page 874).

Styling based on a value in a detail table

Styling rows or cells in a detail table based on a value in the detail table goes a bit different.

First set an ID on the detail table as a whole and create a script that uses `thatID tbody` as the script's selector. If for example the ID is `table_1`, the selector will be: `#table_1 tbody`. Then write a script like the following:

```

for(var i = 0; i < record.tables.detail.length; i++){
    if(record.tables.detail[i].fields['Shipped'] == 1)
        query("tr:nth-child(" + (i+1) + ")", results).css
('color', 'green');
}

```

This script loops over a detail table, evaluating the field `Shipped`. If the value of that field is 1, it looks up the corresponding row in the `results` (the object that contains the selected detail table) and colors the text of that row green. (See also: "query()" on page 926 and "Examples" on page 895.)

Number	Description	Unit Price	Quantity
53674	Thule Crossroad Railing Foot Pack M450	199.95	3
62516	CamelBak Arete 18 Hydration Pack 2016 Black	65.00	1
117653	Swix Carving Kit 1	44.99	1
148080	Swix Alpine Racing Straps Ski Strap	9.99	4

To keep all CSS style rules together you could add the style rules to a class in the CSS file (see "Styling templates with CSS files" on page 553) and assign that class to the a row or cell using `addClass` (see "Examples" on page 883).

For another example, see this how-to: [Change detail line formatting based upon a data field value.](#)

Styling an image

Just like many other elements, images can be given borders and rounded corners, and they can be rotated. How to do this isn't any different from the way it is done with other elements, so it isn't described in this topic, but in general formatting topics; see "Styling and formatting" on page 551.

This topic discusses specific image formatting issues.

Note that image characteristics like brightness and contrast can not be changed within the Designer.

Local formatting vs. style sheets

Just as other elements, images can be styled in two ways:

- With **local formatting**. This means styling the image directly, using the Formatting dialog.
- Via **Cascading Style Sheets (CSS)**. In a style sheet, style rules are declared for elements with different HTML tags, ID's and classes.

See "Styling and formatting" on page 551 for background information about these two methods.

Applying local formatting to an image

To apply **local** formatting to an image, either:

- right-click the image and select **Image...** from the contextual menu
- click the image and select **Format > Image...** from the menu.

Applying style rules to an image

To format an image via a **style sheet**, first give the image an ID or class: select the image, and enter the ID or class on the Attributes pane. This makes it possible to make the CSS style rule target this image specifically, or a set of images with the same class. A style rule with the selector `img` (the HTML image tag) would apply to all images.

Next, create the style rule; see "Styling templates with CSS files" on page 553. Note that when a property isn't present in the style rule editor, it can still be used: click the Advanced button in the style rule editor; enter the property under Property, and its value under Value.

Resizing an image

There are several ways to resize an image after inserting it in the content of a template.



- Click the image and drag the handles to resize it. Press the **Shift** key while dragging, to scale the image proportionally.
- Select the image (see "Selecting an element" on page 469) and type the desired width and height in the respective fields on the **Attributes** pane.
- Select the image and select **Format > Image**, on the menu. On the **Image** tab, change the **width** and **height** of the image.
- Set the size of the image in a style sheet (see "Styling templates with CSS files" on page 553).


The size can be set in a measure or as a percentage of the containing element.

Positioning an image

Wrapping text around an image

Initially, when an image is inserted into a paragraph, it behaves as if it were a character. Text isn't wrapped around an image automatically. To make that happen, you have to change the `float` property of the **image** to `left` or `right`. This anchors the image to the left or right, allowing text to be wrapped around it.

Select the image (see "Selecting an element" on page 469) and use the  (Float left) and  (Float right) icons on the toolbar to change the position of an image within the text.

- The **Float left** button aligns the image to the left. The text is positioned to the right of it and is wrapped around the box.
- The **Float right** button aligns the image to the right, with the text wrapped around it to the left.
- The **No float**  button positions the image where it occurs in the text, as if it were a character. Text is not wrapped around it.

To position an image using the menu, select the image and then select one of the options in **Format > Float**.

Alternatively, open the Formatting dialog (see "Applying local formatting to an image" on page 576): select the image; on the menu, select **Format > Image** and on the **Image** tab, under **Text Wrap**, set the **Float** property.

The `float` property could also be changed via a style sheet. This property isn't present in one of the tab menus of the style rule editor directly, but you can add it and specify its value after clicking the Advanced button in the style rule editor (see "Applying style rules to an image" on page 576).

Pulling an image out of the text flow

When dragged into a template, an image is automatically integrated in the text flow. This means that it will move up or down, depending on the preceding text.

To position the image independently of the text flow, you can change its `position` property to `absolute`. (For an explanation of all possible values for this property, see http://www.w3schools.com/css/css_positioning.asp.)

When an image is placed inside a Box (or Div element), changing its `position` property to `absolute` positions the image absolutely inside that Box.

Note that `float`, the property that can make an image float to the right or left (see "Wrapping text around an image" on the previous page), is a relative positioning property, since it specifies the position of the element relative to its container. This means it is incompatible with the `position: absolute` property.

In the Formatting dialog (see "Applying local formatting to an image" on page 576) the `position` property can be found on the **Image** tab, under **Positioning**.

The `position` property isn't present in one of the tab menus of the style rule editor directly, but you can add it after clicking the Advanced button in the style rule editor (see "Applying style rules to an image" on page 576).

When the `position` property of an element is set to `absolute`, the `top` or `bottom` and `left` or `right` properties can be used to position the element inside its parent with exact values (pixels (px), centimeters (cm), etc). Negative values are allowed.

Note

In Web sections, the `position` property may sometimes be useful for images inside a Div element, but generally elements should not be positioned absolutely. Designs for the Web should be flexible so that they display nicely on a variety of devices and screen sizes.

Background color and/or image

In any type of template, boxes, tables and table cells can have a background color and/or a background image.

To select a background image or color:

1. Right-click the box and click **Box** on the shortcut menu.
2. Alternatively, select the box (see "Selecting an element" on page 469; note that a Box is a `<div>` element) and on the **Format** menu click **Box**.
3. Click the **Background** tab.

To select a **background color**: click the downward pointing arrow next to **Color** to select a color from the list of predefined colors (see "Defining colors, spot colors and tints" on page 583), or click the colored rectangle to open the Color Picker dialog; see "Color Picker" on page 674. In this dialog you can select a color from the color wheel or using the eye dropper tool, set RGB or CMYK color values or enter a hexadecimal color code.

To select a **background image**:

1. Click the **Select Image** button.
2. Click **Resources**, **Disk** or **Url**, depending on where the image is located.
 - **Resources** lists the images that are present in the **Images** folder on the **Resources** pane.
 - **Disk** lists image files that reside in a folder on a hard drive that is accessible from your computer. Click the **Browse** button to select a folder (or an image in a folder). As an alternative it is possible to enter the path manually. The complete syntax is: `file://<host>/<path>`. Note: if the host is "localhost", it can be omitted, resulting in `file:///<path>`, for example: `file:///c:/resources/images/image.jpg`.
 - **Url** lists image files from a specific web address. Select the protocol (**http** or **https**), and then enter a web address (for example, `http://www.mysite.com/images/image.jpg`).
3. With an external image, you can check the option **Save with template**. If this option is checked, the file will be inserted in the **Images** folder on the **Resources** pane. If not saved with the template, the image will remain external. Note that external images need to be available when the template is merged with a record set to generate output,

and that their location should be accessible from the machine on which the template's output is produced. External images are updated (retrieved) at the time the output is generated.

4. Select an image from the list.
5. If the image is contained in a PDF file that consists of more than one page, select the desired page.
6. Click **OK**.
7. Set the size of the image. The options are explained here:
http://www.w3schools.com/cssref/css3_pr_background-size.asp.
8. Set the position of the image in the box.
9. Finally, click **OK**.

Note

It is also possible to set an element's background in a style sheet; see "Styling templates with CSS files" on page 553. When referring to images or fonts from a CSS file, refer to a path that is relative to the current path, which is `css/`. For example: **#header { background-image: url('../images/image.jpg'); }**

Border

In any type of template, boxes, tables and table cells, paragraphs, images and other elements can have a border.

Elements have a rectangular shape, so their border has four sides. Each side of the border can have a different layout.

Adding a border

1. Right-click the element and click the respective element on the shortcut menu. Alternatively, select the element (see "Selecting an element" on page 469) and on the **Format** menu click the respective element.
2. Click the **Border** tab.
3. Uncheck the option **Same for all sides** to be able to style each side of the border separately.

4. Specify the width of the border (side). This is equivalent to the `border-width` property in CSS.
5. Specify the style of the border (side), such as solid, dashed or dotted. This is equivalent to the `border-style` property in CSS.
6. Specify the color of the border (side): click the downward pointing arrow next to **Color** to select a color from the list of predefined colors (see "Defining colors, spot colors and tints" on page 583), or click the colored rectangle to open the Color Picker dialog. In this dialog you can select a color from the color wheel, set RGB or CMYK color values or enter a hexadecimal color code. This setting is equivalent to the `border-color` property in CSS.

Note

It is also possible to set an element's border in a style sheet; see "Styling templates with CSS files" on page 553.

Rounding corners

Any element in a template can have rounded corners. For boxes and images, this option is available in the Formatting dialog. For other elements, you have to create a CSS rule to set the `border-radius` of the element (or class of elements).

Boxes, images and tables

To round the corners of a box, image or table:

1. Select a Box, Image or Table element (see "Selecting an element" on page 469) and on the **Format** menu click the respective element. Alternatively, right-click the element and click the respective element on the shortcut menu.
2. On the first tab in the Formatting dialog (the **Box**, **Image** or **Table** tab respectively) specify the **corner radius** in a measure (10mm, 5px, 0.5in) or percentage (0 - 90%).
3. For a Box or Image, click **Apply** to see the effect without closing the dialog or **OK** to close the dialog.

For a Table, you have to take yet another step. Tables can't have rounded corners and collapsed borders at the same time. All built-in table styles in the Designer have collapsed

borders. For the rounded corners to show, you must create a CSS rule that sets the table's `border-collapse` property to `separate` instead of `collapse`.

1. Click the **Advanced** button at the bottom of the Formatting dialog.
2. Under **Property**, type **border-collapse**.
3. Under **Value**, type **separate**.
4. Add a padding to keep the table cells from sticking out of the rounded corners: under **Property** type **padding** and under **Value** type a measure for the padding.
5. Click OK, and click OK again to close the Formatting dialog.

If the table's rounded corners are still not (fully) visible, check the styles for table cells. Table cells can have their own background color and by that, hide the table's background color - including the rounded corners. Table cells can have rounded corners as well, just as any other elements; see below.

Other elements

To round the corners of elements other than boxes and images, or to have different roundings on different corners, you have to make use of the CSS property: `border-radius`; see http://www.w3schools.com/css/css3_borders.asp.

This is, for example, how you could round the corners of a paragraph:

1. Select the paragraph (see "Selecting an element" on page 469) and then select **Format > Paragraph** on the menu, or right-click the paragraph and select **Paragraph** on the shortcut menu.
2. Click the **Advanced** button at the bottom of the Formatting dialog.
3. Under **Property**, type **border-radius**.
4. Under **Value**, type the value of the corner radius in a measure (10mm, 5px, 0.5in) or percentage (0 - 90%).
5. Click OK, and click OK again to close the Formatting dialog.

Using a CSS file

Of course you could also add this rule to a CSS file; see "Styling templates with CSS files" on page 553. The following rule sets the border-radius of the corners of all paragraphs to 5 pixels:

```
p { border-radius: 5px; }.
```

To make this rule apply to one specific paragraph, first give the paragraph an ID (select the

paragraph and type the **ID**, for example **rounded**, on the **Attributes** pane). Then add the ID to the selector of the CSS rule, for example `p#rounded { border-radius: 5px; }`.

To make the CSS rule apply to a set of paragraphs with the same class, first give the paragraphs the same **class** (for example **rounded**). Then add that class to the selector of the CSS rule, for example `p.rounded { border-radius: 5px; }`.

Colors



Colors make an important contribution to the look and feel of your templates. This topic explains how to define and apply colors and how to keep them consistent in different output channels.

Defining colors, spot colors and tints

Color selectors, such as the drop-down list on the toolbar, initially contain a small set of colors. Add your own colors so that they can be used throughout the templates, in all contexts and in color selector dialogs as well as with their names in style rules (see "Styling and formatting" on page 551).

Defining colors

To define colors:

1. Select **Edit > Colors** on the menu.
2. Add a color. There are two ways to do this:
 - Click the **New** button (the green plus).
 - Select an existing color from the list and copy it using the **Duplicate** button . (The Filter drop-down limits the list to colors of a certain type.) Select the new color and click the **Edit** button .
3. In the Edit color dialog, type a name for the color (or let the Designer create a name based on the values that you select). The color's name can be used in style sheets. It should not contain spaces or special characters.

Tip

Working with style sheets? Choose a name that informs about the purpose of the color, rather than a name that describes the color. This way you won't have to change the color's name in the style sheets when you change the color.

4. Click **Color**. (Tint is used for transparent colors.)
5. Select the color type: **CMYK** or **RGB**.
The letters **CMYK** stand for Cyan (a greenish-blue color), Magenta (reddish-purple), Yellow and Key (black). In color printing, these are the usual primary colors.
RGB stands for Red, Green and Blue. In the RGB color model, red, green, and blue light are added together in various ways to reproduce a wide range of colors. This model is typically used for electronic devices.
For information about the **Spot color** and **Overprint** options see "Defining a spot color" below.
6. Drag the slider bars to set the values for the color and click OK or Apply.

Defining a spot color

A **spot color** is any color generated by an ink (pure or mixed). Note that spot colors can only be used on certain printers.

If your printer can use spot colors and you want a spot color to be used in a Print context, define the color as described above, making sure to:

- Match the color's **name** to that of the spot color used in the printer.
- Check the option **Spot color**.
- If applicable, check the **Overprint** option for this spot color. Overprinting refers to the process of printing one color on top of other colors. This is sometimes required, for example to deal with special print applications, such as applying UV ink or varnish to a certain area, or to avoid mis-registration when printing black on top of coloured areas.

Defining a tint

A tint is a transparent color, based on another color in the template. To define a tint:

1. Select **Edit > Colors** on the menu.
2. Click the **New** button (the green plus) to add the tint.

3. From the Type drop-down, select **Tint**.
4. In the Edit color dialog, type a name for the color (or let the Designer create a name based on the values that you select). The color's name can be used in style sheets. This name should not contain spaces or special characters.
5. Select one of the existing colors in the template as t the **Source** of the color. The tint or opacity will be applied to this color.
6. Check **Use opacity** if you want to set the Tint slider to use Opacity instead.
7. Use the slider to set the percentage of the tint or opacity, or type the percentage directly in the input box and finally click OK.

Applying a color

Colors can be applied to elements in your templates locally or through style sheets.

Using colors in style sheets

It is highly recommended to use style sheets in templates right from the start. Even more so if the communications are going to be output to different output channels, or if they consist of different sections (for example, a covering letter followed by a policy). Using CSS with templates allows a consistent look and feel to be applied. A style sheet can change the look of multiple elements, making it unnecessary to format each and every element in the template, time and again, when the company's layout preferences change. See "Styling templates with CSS files" on page 553.

In style sheets, you can color every type of element that has a CSS color property, such as **color**, **background-color** or **border-color**. Use the color's name as it is defined in the Designer, or any legal color value: a valid color name (see [color names on w3schools](#)), hexadecimal color code (see [w3school's color picker](#)), RGB color value, for example `rgb(216,255,170)` or CMYK color value, for example `cmyk(15%, 0%, 33%, 0%)`.

The following CSS rule applies `MyColor`, which is a custom color (see "Defining colors, spot colors and tints" on page 583), to the text of all paragraphs:

```
p {  
color: MyColor;  
}
```

CMYK colors

You may use the custom `cmyk()` CSS function to assign a CMYK color to any element, or a series of elements. The following example assigns a steel blue color as a background for all H1

elements:

```
h1 {  
background-color: cmyk(33%, 17%, 0%, 20%);  
}
```

Coloring text

Instead of using a style sheet (see above), you can color text locally:

1. Select text or an HTML element that contains text (see "Selecting an element" on page 469).
2. On the menu, select **Format > Color**, or click the black triangle on the **Text color** toolbar button.
3. Select one of the colors in the list, or click **Other** to set all aspects of the text style, including text color and/or background color.

Coloring backgrounds and borders

Instead of using a style sheet (see above), you can color a background or border locally. This is how:

1. Select an HTML element (see "Selecting an element" on page 469).
2. On the **Format** menu, click the element. For a **div** element, click **Box**. The Formatting dialog opens up.
3. Click the **Border** or **Background** tab.
4. Click the downward pointing arrow next to **Color** to select a color from the list of predefined colors (see "Defining colors, spot colors and tints" on page 583).
Alternatively, click the small rectangle to the right of the color list to open the Color Picker dialog. In this dialog you can select a color from the color wheel. You can also choose the color mode: RGB or CMYK. For an explanation of these two modes, see "Defining colors, spot colors and tints" on page 583; for an explanation of the other options in this dialog, see "Color Picker" on page 674.
You could also type a name or value in the Color field directly. It must be a valid color name (see [color names on w3schools](#)), a hexadecimal color code (see [w3school's color picker](#)), RGB color value, for example `rgb(216, 255, 170)` or CMYK color value, for example `cmyk(15%, 0%, 33%, 0%)`.
5. Click **OK** or **Apply**.

Color management

Color profiles can keep colors consistent across different outputs. To manage color profiles, select **Edit > Color settings**; for an explanation of the options in the Color settings dialog, see "Color Settings" on page 676.

Fonts

In templates for personalized customer communications you can use the fonts that are provided with the Designer, as well as imported fonts.

Note

Hosting non-standard fonts on a Windows Server operating system (as opposed to importing them into the template) is not recommended.

If output is produced on the server whilst running under a different account, that account might not have access to the font.

If you do add a font to Windows Server, do not forget to restart the machine, as otherwise the font might not be available.

Applying a font

To apply a particular font to a piece of text, you can:

- Select some text, or an element that contains text (see: "Selecting an element" on page 469) and select a font from the **Fonts** drop-down on the toolbar.
- Use the name of the font in a CSS rule, for example:

```
body {  
font-family: Verdana, Arial, sans-serif;  
}
```

Instead of the body tag, any element that can have the CSS property 'font-family' can be used.

Make sure that the rule is applied to the text that you wanted to apply the font to; see "Step 2: apply CSS to the content" on page 558.

Note

The reason for specifying more than one font in a style sheet for web pages and emails is that the font might not be available on the device on which they are viewed.

Order the font names by preference. The last one should be the generic font family (either serif or sans-serif).

Importing a font

To import a font into a template:

- Drag the appropriate font files into the **Fonts** folder on the **Resources** pane.

When text is displayed in an imported font, the Designer can mimic the bold and italic versions of that font. If you have separate files for the bold, italic and possibly other versions of a font, you can make the Designer use the appropriate files to style text. To do this:

1. Import the files for the bold, italic and/other versions of the font into the Fonts folder.
2. On the **Edit** menu, click **Fonts**, to open the Font Manager.
3. Select the normal version of the imported font and duplicate it using the **Duplicate** button, once for each version of the font.
4. For each of the duplicates, combine a font effect with a file:
 - Click a duplicate and click the button Edit. Note: don't change the duplicate's name!
 - Select the appropriate font effect (font-weight and/or font-style).
 - Check the file or files the Designer should use for that effect. Per file type, one file can be checked.
5. Close the Font Manager.

The Designer currently supports 4 font types: TTF, OTF, WOFF, EOT and SVG.

When you are creating a **Web** template, keep in mind that the different font types are not supported by all clients; for instance, EOT and SVG are used only by Explorer and Safari, respectively.

When creating an **Email** template, it's better to import several types of the same font, in order for any client to see the appropriate fonts.

In the case of a **Print** context you do not need to provide alternative fonts, because the output is not displayed using a font from the device on which the output is read.

Note

Font software may have specific restrictions for copying and redistribution. Please consult the license agreement for each font vendor before using it in a template. It is your responsibility to comply with the requirements of third-party agreements.

Applying an imported font

Once a font is imported, it is automatically added to the Fonts drop-down on the toolbar.

It can also be used in the style sheets, even in combination with other fonts, for example:

```
body {  
font-family: 'MyWebFont', Arial, sans-serif;  
}
```

Using remote fonts

In order to use a remote font, you have to add a remote style sheet that points to a web font style sheet, for example <https://fonts.googleapis.com/css?family=Roboto+Slab>. For instructions see "Using a remote style sheet" on page 555.

Remote fonts can be applied to content in a Master page, section, or Snippet. They may be used in a style sheet and they are automatically added to the Fonts drop-down on the toolbar. Note that the list of font names is based on the style sheets that are included in the active section (see "Applying a style sheet to a section" on page 559) or, when editing a Snippet, in the section that was active when the Snippet was opened.

Note

Support for remote fonts in email clients cannot be relied upon, and not all remote fonts are supported by all browsers. It is therefore recommended to add fallback fonts to the specific style

rules whenever using remote fonts in a **Web** or **Email** section (see "Applying a font" on page 587).

Locale

The locale is a setting that can affect date, time and currency output, and other formatting that depends on location and language. This setting is specific to each template, so changing it for one template will not affect other templates.

Assume that a record set has a `Date` field that contains the following date: 4/11/12, and that this field has been added to the template using the Text Script Wizard with the Long Date format (see "Using the Text Script Wizard" on page 607 and "Formatting variable data" on page 610). If the locale is set to `en-US`, the date appears on the page as **April 1, 2016**. Setting the locale to `fr-CA` makes this text appear as **1 avril 2016**. Setting it to `zh-CN` will print **2016年4月1日**.

The locale can also be used in scripts; see "Writing your own scripts" on page 624 and "Designer Script API" on page 874.

Changing the locale

By default, the locale is the same as the operating system's locale setting. To change this setting for the currently open template:

1. On the menu, select **Edit > Locale**.
2. Use the drop-down to select how the locale is to be set for the current template:
 - Select **System Locale** to use the operating system's locale settings. The operating system's locale is set in the **Region** settings of the control panel. Note that when output is generated on a different operating system, that operating system's locale will be used.
 - Select **Explicit Locale** to specify a static locale which will remain static for this template, whichever server the template is used on. Use the **Locale** drop-down to select a specific locale. The locales comprise a language code followed by a 2-letter country code (`de-DE`, `zh-CN`, `fr-CA`, `fr-FR`, etc), as defined by the international standards ISO-639-1 and ISO 3166.
 - Select **Data Field** to use a data field from the record. The locale will be record-specific in this case. Use the drop-down to select a field within the current Data

Model that contains the locale. This field must be a string and contain the exact locale to be used, such as "en" or "fr-CA". It cannot be an alias such as "english" or "french". The locale supports language codes (en, fr, etc), as well as language codes followed by a 2-letter country code (de-DE, zh-CN, fr-CA, fr-FR, etc). The language codes are defined by ISO-639-1. The 2-letter country code as defined by ISO 3166.

3. Click **OK** to apply the setting. The setting will be saved with the template.

Spacing

Boxes, tables, paragraphs and many other elements have a **margin** and **padding**.

The margin is the white space around an element, outside the border. It is used to position an element in relation to the other elements, by putting more space between the element and its surrounding elements.

The padding is the space between an element's content and its border. It is used to position the content of the element inside the border.

Elements have a rectangular shape, so they have four sides. The margin and padding have be different on all sides.

Tip

Use a negative left margin to create a hanging paragraph or image.

To set the spacing:

1. Right-click the element and click the respective element on the shortcut menu. Alternatively, select the element (see "Selecting an element" on page 469) and on the **Format** menu click the respective element.
2. Click the **Spacing** tab.

Note

All settings in the Formatting dialog are in fact CSS style rules. Click the **Advanced**

button to manually add CSS properties (at the left) and values (at the right). For more information about CSS, see "Styling and formatting" on page 551.

It is also possible to change an element's formatting via a style sheet; see "Styling templates with CSS files" on page 553.

3. Set the value for the **padding** in measure or percentage. You can do this for each side separately, which is equivalent to the **padding-top**, **padding-bottom**, **padding-left** or **padding-right** property in CSS. To set the same padding for all sides, check the option **Same for all sides**. This is equivalent to the **padding** property in CSS.
4. Set the value for the margin in measure or percentage. You can do this for each side separately, which is equivalent to the **margin-top**, **margin-bottom**, **margin-left** or **margin-right** property in CSS. To set the same margin for all sides, check the option **Same for all sides**. This is equivalent to the **margin** property in CSS.
5. Click **OK**, or click **Apply** to apply the changes without closing the dialog.

Personalizing Content

Variable-data printing is a form of digital printing in which elements such as text and graphics may be changed using information from a database or data file. It prints unique documents with customized messages for each customer. This is exactly what you can do with Connect: using variable data you can personalize your company's communications.

Before you can start personalizing the content of a template, you must open a Data Mapping Configuration, data file or database; see: "Loading data" on page 594.

The most common ways to personalize templates are listed below.

Variable data

Variable data are data from a database or data file that are used to personalize documents for each customer. Variable data fields can be inserted in the text directly. For example, if a person's last name can be found in your data, the field that holds the last name can be used in the text of a web page, letter or email. Scripts in PlanetPress Connect Designer are the basis of Variable Data Printing.

The easiest, quickest and most direct way to add customer data to content is via drag and drop; see "Variable Data" on page 604.

The drag-and-drop method results in a Text Script. Another way to create a Text Script is to use the Text Script Wizard. Often it is better to use the Text Script Wizard than the drag-and-drop method.

The Text Script Wizard gives you more control over the way data is displayed. It can insert one or more data fields, each with an optional prefix and suffix. For blocks of data, such as addresses, the Text Script Wizard definitely is the better choice. See "Using the Text Script Wizard" on page 607.

Conditional content

In a template you may want to reveal content - text or images - to one group of recipients, but hide it from others. You can use a Conditional Script Wizard to achieve this, if you have a data field in your data on the basis of which a condition can be set. See "Showing content conditionally" on page 613.

Conditional Print sections

Entire Print sections can be included in or omitted from the output on the basis of one or more values in variable data. See "Conditional Print sections" on page 616.

Dynamic images

Dynamic Images are dynamic in the sense that they are replaced by another image when a data field contains a certain value. Think of a signature image being swapped based on the sender's name, for example. You can use the Dynamic Image Script Wizard to make this happen; see "Dynamic Images" on page 617.

Dynamic tables

A Dynamic Table is a table with a variable number of rows that can overflow on one or more pages. It can also display subtotals on transport lines. In invoices, a Dynamic Table is an essential element. Read "Dynamic table" on page 618 to learn how to insert a dynamic table.

Snippets

Snippets are pieces of content that can be re-used within the same template, in all contexts and sections. Snippets can contain any contents that a section can have, such as text, images, variable data, dynamic tables, etc. They are often very useful to personalize content, especially in combination with variable data and scripts. See "Snippets" on page 548 and "Loading a snippet via a script" on page 640.

Scripts

Self-made scripts

As soon as you want to do more than what can be done with the available (Text, Conditional) Script Wizards, self-made scripts are the solution. You could, for example, combine data of two or more data fields in a condition for conditional text. Or you could load a part of a snippet depending on the value of a data field. With a self-made script you can achieve anything that can be done by any of the Script Wizards, and much more. For an introduction on this, see "Writing your own scripts" on page 624.

Control Scripts

When output is generated from a template, Control Scripts run **before** all other scripts, when a record is merged with a context. They determine how different sections of the context are handled. They can, for example, make the page numbering continue over all Print sections, split Email attachments, or omit Print sections from the output.

Some knowledge of JavaScript is needed to edit Control Scripts, just as for any other self-made scripts, because there is no Control Script Wizard; see "Writing your own scripts" on page 624.

See "Control Scripts" on page 645.

Loading data

Before you can add variable data fields to a template in the Designer, you need to have a Data Model and a sample of customer data. At the design stage the Designer doesn't have to have access to all data; it just needs to know which data fields exist in your data and it needs some data to be able to display a preview of the output.

To get access to a Data Model and data, you can open:

- a Data Mapping Configuration, see "Loading a Data Mapping Configuration" on the facing page
- a data file, see "Adding data from a data file" on page 597
- a database, see "Adding data from a database" on page 599.

A Data Model and sample data are part of a Data Mapping Configuration.

When you open a data file or a database, the Data Model will be derived from it **unless** there already is an open Data Mapping Configuration; in that case, the current Data Mapping Configuration will try to retrieve data from the file or database, using its own Data Model and extraction logic.

After opening a Data Mapping Configuration or opening a data file or database, the **Data Model** pane at the right hand bottom shows the data fields that occur in the data.

The **Value** column displays data from the first record in the data file. Use the **First**, **Previous**, **Next** and **Last** buttons to browse through the records, or use the Page Up, Page Down, Home and End keys.

Note

Although it is possible to load data from a data file or database in the Designer without creating a Data Mapping Configuration for it, generally the best way to extract data is by creating a Data Mapping Configuration. With a Data Mapping Configuration you can, among other things:

- Use the same data file with a different template, or use different kinds of data files with the same template.
- Load transactional or structured data. If there are detail lines, transactions, or any variable number of items to put into the template, you need to a Data Mapping Configuration to extract them.
- Format, transform, conditionally include/exclude and enhance data from the source file.
- Use Workflow to automate the extraction of data from this kind of data file.


Tip

If you have no data at hand, download a demo from <http://demo.objectiflune.com> and open a dummy data file to test with.

Loading a Data Mapping Configuration

If you have used the DataMapper first, you probably already have an open Data Mapping Configuration. Its Data Model and sample data will automatically be used when you start creating a template. You might have to click the **Synchronize Model** button on the **Data Model** pane, to update the fields.

To open a Data Mapping Configuration:

1. Open the Welcome screen: click the Home  icon at the top right or select **Help > Welcome** on the menu.
2. Click **Open an existing configuration**.
3. Select the Data Mapping Configuration and open it.
4. At the top of the workspace, click the tab with the name of the template's section to go back to the template.
5. Click the button **Synchronize model** at the top of the **Data Model** pane to reload the data model.

Note

The EXTRADATA field that appears as the first field in each record and in each detail table is automatically added to the Data Model by the DataMapper. It offers the possibility to add extra data to an existing Data Model, for example when Workflow has to perform a lookup to retrieve a value from a database and add that value to a new field in the Data Model.

The EXTRADATA field can be added to the template just like any other data field (see "Variable Data" on page 604). When it contains a JSON string, this value can be read with a script (see "loadjson()" on page 922).

Note

When generating output with just a Data Mapping Configuration, the template is merged with the

complete sample data file that is part of the Data Mapping Configuration. The output is **not** limited to the number of records shown in the Data Model pane (which is one of the settings in the DataMapper).

Adding data from a data file

1. Click **File**, select **Add Data** and then click **From file data source**. Browse to the location of the file and select it.

The Designer can open the following types of data files:

- Tabular files: CSV files (.csv) and Excel files (.xls, .xlsx)

Note

Excel files saved in "Strict Open XML" format are not supported yet.

- Microsoft Access Database (.mdb, .accddb)
 - XML files (.XML)
 - PDF/VT files
2. Review the options presented, to ensure that the data will be interpreted correctly. The options available depend on the type of data file (see below).

Excel (XLS/XLSX) file options

- **First row contains field names:** Check this option to use the first line of the CSV as headers. This option automatically names all extracted fields.
- **Sheet:** Only one sheet can be selected as the data source.

CSV file options

- **Encoding:** The Designer can not infer from a CSV file what encoding it is in. The default is right in the large majority of cases, but when it isn't, it can be very difficult to figure out the correct encoding. Ask your source what the encoding of the file is.
- **Field separator:** Choose the character that separates the fields in the file.

- **Comment delimiter:** If there are comment lines in the file, type the character that starts a comment line.
- **Text Delimiter:** Type the character that surrounds text fields in the file. Other delimiters will not be interpreted within these text delimiters.
- **Ignore unparseable lines:** When checked, any line that does not correspond to the above settings will be ignored.
- **First row contains field names:** Check this option to use the first line of the CSV as headers. This option automatically names all extracted fields.

MDB file options

- **File:** Include the full path to the file.
- **Password:** If the file isn't password protected, you can click **Next** without filling out this field.
- **Table name:** Use the drop-down to select the appropriate table or stored query to retrieve the appropriate data set.
- **Encoding:** Use the drop-down to select the encoding with which to read the data in the table.

XML File options

Select what level of XML elements defines a record.

The **Trigger** is what triggers the creation of a new record. It can be set to:

- **On element:** This defines a new record when a new element occurs on the selected XML level.
- **On change:** This defines a new record when a specific field under the chosen XML level has a new value. After selecting this option, you have to select the field that triggers the creation of a new record.

PDF/VT file options

After selecting a file, use the drop-down to select what level in the PDF/VT file defines a record in your data. The names of the levels are taken from the PDF/VT file itself. (See "About PDF/VT files" on the next page.)

All metadata fields that belong to the chosen level and levels higher up in the tree structure will

be listed. The lower the chosen level is in the tree structure, the more records you will get and the more metadata fields will appear in the list.

Select metadata fields to add them to your data. Their property names will be used as field names in the Designer's data model.

About PDV/VT files

The pages in PDF/VT files can be grouped on several levels. PDF/VT files can have a variable number of levels in their tree structure. The level's names are variable as well, with the exception of the lowest level, which is always called the **page** level. Metadata can be attached to each level in the structure.

AFP file options

After selecting a file, use the drop-down to select what level in the AFP file defines a record in your data. The levels are defined in the AFP file itself. (See "About AFP files" below.)

All metadata fields that belong to the chosen level and higher levels in the tree structure will be listed. The lower the chosen level is in the tree structure, the more records you will get and the more metadata fields will appear in the list.

Select metadata fields to add them to your data. Their property names will be used as field names in the Designer's data model.

About AFP files

Pages in AFP files are arranged in a tree structure, comprising one **document** at the top of the structure, **pages** at the bottom of the structure and one or more levels of **page groups** in between. (Unlike in PDF/VT files, the names of levels in AFP files can not be chosen freely.) In other words, an AFP file always consists of one document, that can contain page groups, of which each can be divided in page groups, and so on; the page groups at the lowest level contain pages.

Metadata can be attached to each level in the structure.

Adding data from a database

1. Click **File**, select **Add Data** and then click **From database data source**. Browse to the location of the file and select it.

The Designer can open databases from the following types of data sources:

- MySQL
- Microsoft Access Database (.mdb, .accddb)
- SQL Server

- ODBC DataSource
 - JDBC
 - Oracle.
2. Review the options presented. The options available depend on the type of database data source; see below.

MySQL

1. Enter the appropriate information to connect to the database:
 - **Server:** Enter the server address for the MySQL database.
 - **Port:** Enter the port to communicate with the MySQL server. The default port is *3306*.
 - **Database name:** Enter the exact name of the database from where the data should be extracted.
 - **User name:** Enter a user name that has access to the MySQL server and specified database. The user only requires *Read* access to the database.
 - **Password:** Enter the password that matches the username above.
2. Click **Next** and enter the information for the source table.
 - **Connection string:** Displays the full path to the database.
 - **Table:** Use the drop-down to select the appropriate table or stored query to retrieve the appropriate data set.
 - **Encoding:** Use the drop-down to select the encoding with which to read the data in the table.
3. Click **Finish** to open the database.

Microsoft Access

1. Enter the appropriate information to connect to the database:
 - **File name:** Browse to your Microsoft Access database file (.mdb)
 - **Password:** Enter a password if one is required.

2. Click **Next** and enter the information for the source table.
 - **Connection string**: Displays the full path to the database.
 - **Table**: Use the drop-down to select the appropriate table or stored query to retrieve the appropriate data set.
 - **Encoding**: Use the drop-down to select the encoding with which to read the data in the table.
3. Click **Finish** to open the database.

SQL Server

1. Enter the appropriate information to connect to the database:
 - **Server**: Enter the server address for the SQLServer database.
 - **Port**: Enter the port to communicate with the SQLServer. The default port is *1433*.
 - **Database name**: Enter the exact name of the database from where the data should be extracted.
 - **User name**: Enter a username that has access to the SQLServer and specified database. The user only requires *Read* access to the database.
 - **Password**: Enter the password that matches the username above.
2. Click **Next** and enter the information for the source table.
 - **Connection string**: Displays the full path to the database.
 - **Table**: Use the drop-down to select the appropriate table or stored query to retrieve the appropriate data set.
 - **Encoding**: Use the drop-down to select the encoding with which to read the data in the table.
3. Click **Finish** to open the database.

ODBC DataSource

1. Select the ODBC system data source. Note: Only 32-bit data sources are currently shown in this dialog, even if your system is 64-bits.

2. Click **Next** and enter the information for the source table.
 - **Connection string**: Displays the full path to the database.
 - **Table**: Use the drop-down to select the appropriate table or stored query to retrieve the appropriate data set.
 - **Encoding**: Use the drop-down to select the encoding with which to read the data in the table.
3. Click **Finish** to open the database

JDBC

1. Enter the appropriate information to connect to the database:
 - **JDBC Driver**: Use the drop-down to select which JDBC Driver to use for the database connection.
 - **JAR file path**: Enter a path to the JAR file that contains the appropriate driver for the database below.
 - **Server**: Enter the server address for the database server.
 - **Port**: Enter the port to communicate with the server.
 - **Database name**: Enter the exact name of the database from where the data should be extracted.
 - **User name**: Enter a username that has access to the server and specified database. The user only requires *Read* access to the database.
 - **Password**: Enter the password that matches the username above.
 - **Advanced mode**: check to enable the Connection String to manually enter the database connection string.
 - **Connection string**: Type or copy in your connection string.
2. Click **Next** and enter the information for the source table.
 - **Connection string**: Displays the full path to the database.
 - **Table**: Use the drop-down to select the appropriate table or stored query to retrieve the appropriate data set.
 - **Encoding**: Use the drop-down to select the encoding with which to read the data in the table.
3. Click **Finish** to open the database.

Oracle

1. Enter the appropriate information to connect to the database:
 - **Server:** Enter the server address for the Oracle database.
 - **Port:** Enter the port to communicate with the Oracle server.
 - **Database name:** Enter the exact name of the database from where the data should be extracted.
 - **User name:** Enter a username that has access to the Oracle server and specified database. The user only requires *Read* access to the database.
 - **Password:** Enter the password that matches the username above.
2. Click **Next** and enter the information for the source table.
 - **Connection string:** Displays the full path to the database.
 - **Table:** Use the drop-down to select the appropriate table or stored query to retrieve the appropriate data set.
 - **Encoding:** Use the drop-down to select the encoding with which to read the data in the table.
3. Click **Finish** to open the database.

After adding data from a database, the **Data Model** pane at the right hand bottom shows the data fields that occur in the data.

The **Value** column displays data from the first record in the data file. Use the **First**, **Previous**, **Next** and **Last** buttons to browse through the records.

Add a counter using the Generate Counter Wizard

Generating a counter is useful for numbered tickets or any other template requiring sequential numbers but no variable data.

The Generate Counter Wizard creates a record set with a Counter field and in that field, the current counter value for each record. The Counter starts and stops at set values and is incremented by a set value as well.

1. To open the Generate Counter Wizard, select **File > Add data > Generate counters**.
2. Adjust the settings:
 - **Starting value:** The starting number for the counter. Defaults to 1.
 - **Increment value:** The value by which to increment the counter for each record. For example, an increment value of 3 and starting value of 1 would give the counter values of 1, 4, 7, 10, [...]
 - **Number of records:** The total number of counter records to generate. This is not the end value but rather the total number of actual records to generate.
 - **Padding character:** Which character to add if the counter's value is smaller than the width.
 - **Width:** The number of digits the counter will have (prefix and suffix not included). If the width is larger than the current counter value, the padding character will be used on the left of the counter value, until the width is equal to the set value. For example for a counter value of "15", a width of "4" and padding character of "0", the value will become "0015".
 - **Prefix:** String to add before the counter, for example, adding # to get #00001. The prefix length is not counted in the width.
 - **Suffix:** String to add after the counter. The suffix length is not counted in the width.
3. Click **Finish** to generate the Counter record set.

Tip

While the Generate Counter script is really useful for things like raffle tickets, it's unusable in combination with a data file or database, as it cannot complement that data automatically. This can only be done with a script. A script that adds a counter to data, using the current record index to calculate the current counter value, can be found in this how-to: [Manual counter in designer](#).

Variable Data

Variable data are data from a database or data file that are used to personalize documents for each customer. Variable data fields can be inserted in the text directly. For example, if a person's last name can be found in your data, the field that holds the last name can be used in the text of a web page, letter or email. Scripts in PlanetPress Connect Designer are the basis of Variable Data Printing.

After loading a Data Mapping Configuration or data from a data file or database (see "Loading data" on page 594), you can add variable data fields to the contents of your template. You can do this via the drag-and-drop method, or using the Text Script Wizard.

Use the **Text Script Wizard** when there are empty fields in the data, and the value of a data field needs to be preceded or followed by a space, line break or text in the template. Otherwise, empty data fields will cause empty lines and superfluous white spaces to show up in the text. You should also use this method for blocks of data, such as address blocks, and when you want to format data differently, for example, when you want a number to be displayed as a currency.

You can use the **drag-and-drop** method for simple fields that do not need to be preceded or followed by a space, line break or text.

Note

Web templates are personalized just like any other template. There are a few extra possibilities, though; see "Using variable data on a Web page" on page 389.

Inserting variable data directly (drag-and-drop)

An easy, quick and direct way to insert variable data in the content is via drag and drop:

1. Open the section you want to add the data field to.
2. Drag and drop a data field from the **Data Model** pane at the bottom right into the content of your template.
To select and insert multiple data fields at the same time, press **Shift** or **Ctrl**, whilst selecting fields in the **Data Model** pane.

What happens is that:

- A **placeholder** for the value of the data field shows up in the text. It looks as follows: @FIELDNAME@.
- A **text script** appears in the **Scripts** pane at the bottom left.

A **text script** replaces placeholders in the content with the value of a data field in the current record.

Switch to the **Preview** tab at the bottom of the workspace to see the script in operation. The value of the corresponding data field in the first record appears instead of the placeholder, everywhere where the placeholder is found in the text. This value will be refreshed when you browse through the records in the Data Model pane.

When the output (the letter, email, etc.) is generated, the text script executes for each record in the record set, and each time it replaces the placeholders by the value of the field in the current record.

In the **Scripts** pane you can see that the script has a **name** and a **selector**.

The drag-and-drop method automatically generates a script that is named after the data field (see the first column of the **Scripts** pane).

The **selector** (in the second column in the **Scripts** pane) is the text that the script will replace. The selector that the drag-and-drop method generates for a script, is the same as the placeholder that is placed in the text.

When you drag the same field to the content again, a second placeholder appears in the text, but no new script is added. The existing script will find and replace all placeholders that match its selector.

Tip

Press the **Alt** key while dragging, to wrap the placeholder in a **span**, give the span an **ID** and have that ID used as the script's selector.

Press the **Ctrl** key while dragging, to wrap the placeholder in an absolute positioned box (a **div**) at the cursor position. A unique ID is assigned to the box and used as the script's selector. This method is particularly useful when the document mainly consists of a PDF used as the background image of a section (see "Using a PDF file as background image" on page 339).

Tip

Drag the data field directly to the **Scripts** pane to create a script without adding a placeholder to the template.

Note

Looking for text in a text is a less optimized operation and may impact output speeds in longer documents. To speed up the output process, put the placeholder(s) in a Box or Span (see "Boxes" on page 513), give that Box or Span an ID and use that ID as the script's selector. See "Using the Text Script Wizard" below for an explanation about the various types of selectors. For more tips to make a template generate output faster, see "Optimizing scripts" on page 636.

Using the Text Script Wizard

The Text Script Wizard can insert one or more data fields into your template, each with an optional prefix and suffix. It is recommended to use the Text Script Wizard for blocks of data, such as address blocks, and when data fields can be empty or need to be formatted differently.

1. Create a new text script and open the Text Script Wizard. There are two ways to do this:
 - On the **Scripts** pane at the bottom left, click the black triangle on the **New** button and click **New Text Script**. A new script appears in the list. Double-click the new script to open it.
 - Select a word in the content. Right-click the selection and on the shortcut menu, choose **Text Script**.

The Text Script Wizard appears.

2. Change the name of the script to make clear what it does.
3. The **selector** states the text to be found in the template. The results can be replaced by the script.

Tip

Hover over the name of a script in the **Scripts** pane to highlight parts of the template that are affected by the script.

- **Text**, for example: @lastname@, or {sender}. The text doesn't have to have any special characters, but special characters do make it easier to recognize the text for yourself. In the Text Script Wizard, click **Text** and type the text to find.

Note

A script made with the Text Script Wizard for a block of data already runs faster than a series of individual scripts, because it only has one selector. However, searching for text can be a lengthy operation, compared to searching for an element with an ID. When speed matters, select one of the two remaining options: **Selector** or **Selector and Text**. See also: "Testing scripts" on page 632 and "Optimizing scripts" on page 636.

- An **HTML/CSS selector**:

- HTML elements, such as a paragraph. In the Text Script Wizard, click **Selector** and type the HTML tag without the angle brackets, for example: **p**.
- HTML elements with a specific class. In the Text Script Wizard, click **Selector** and type the class name, including the preceding dot, for example: **p.green** for all paragraphs with the class 'green' or **.green** for all kinds of HTML elements that have the class 'green'. See "Styling and formatting" on page 551 for an explanation about CSS (Cascading Style Sheets).
- An HTML element with a specific ID. In the Script Wizard, click **Selector** and type the ID, including the preceding #, for example: **#intro**.

Note

Each ID should be unique. An ID can be used once in each section.

- Etcetera. See http://www.w3schools.com/cssref/css_selectors.asp for more selectors and combinations of selectors.
- A **selector and text**. This is text inside an HTML element (or several HTML elements) with a specific HTML tag, class or ID. In the Text Script Wizard, click **Selector and text** and type the selector and the text in the respective fields.
4. Click the the downward pointing arrow in the first row in the column **Field**. Select a data field from the list that appears.
 5. Add a **Prefix** and/or a **Suffix**. The prefix and suffix can contain text and/or HTML tags. If a field is empty, the prefix and suffix will be ignored, which means you can add line returns

and static text, such as:

- with a Number field, Prefix: Your invoice (one space at the end), Suffix: is now ready to be viewed!
- with a field LastName, Suffix `
` (which adds a line break)
- with a field State, Prefix: , (comma then space).

For a comma between fields, use the Prefix of the second field, if you don't want a comma when the second field has no value.

6. The Wizard allows you to reformat the data (for example, apply uppercase, apply thousand separators to numbers, etc.). Click the column **Format**, click the downward pointing arrow and select one of the formats. See "Formatting variable data" on the facing page.
7. Add as many data fields as you need, following the same procedure.
8. Optionally, you can click **Options** to specify where and how the script inserts its results:
 - As **HTML**. HTML elements in the results are processed and displayed as HTML elements. For instance, `this is bold` will be displayed as **this is bold**. This is the default setting.
 - As **text**. This inserts the results as-is, meaning HTML tags and elements are displayed as text in the output. In this scenario, "`
`" shows up in the text and does not insert a line break.
 - As the value of an **attribute** of an HTML element. The selector of the script should be an HTML element. Which attributes are available depends on the selected HTML element. If the script's selector is an image (`` element) for example, and the attribute is `src`, the script will modify the image's source. The script's results should be a valid value for the chosen attribute.

Note

When checked, the option **Convert fields to JSON string** writes the results from the script into an attribute or text as a JSON string. This is useful for web contexts where a front-end script can read this value easily.

9. Close the Text Script Wizard and type the placeholder for the results of the script in the content of your template, or make sure that there is at least one element that matches the selector of the script.

10. Hover over the name of the script in the **Scripts** pane. In the workspace you will see which parts of the template are affected by the script. If the script produces an error, the error message will be displayed in a hint on the **Scripts** pane.

Tip

When one of the included data fields is empty, the respective line, including the prefix and suffix, is skipped. The result of the script will be shorter, causing the rest of the content to move up or down. If, in a Print context, you don't want the result of the script to be part of the text flow (for example, when a letter is going to be sent in an envelope with a window), put the placeholder for the script in a positioned box (see "Boxes" on page 513 and "How to position elements" on page 567).

Tip

- An example of how to create an address block using the Text Script Wizard is described in a how-to; see [How to create an Address Block](#).
- To use only part of a data field, or to split the data, you will have to write a script. For an example, see this How-to: [How to split a string into elements](#).

Formatting variable data

When a Text Script, made with the Text Script Wizard (see "Using the Text Script Wizard" on page 607) adds variable data to a template, it can easily change the way the data are formatted as well. This is done in the Text Script Wizard through a special formatting modifier or a format mask for each field that the script adds to the template.

The available formatting functions depend on the data type of the corresponding field in the Data Model. In a Data Mapping Configuration you can set the data type of each field. When you open a data file or database without a Data Mapping Configuration, all fields are text fields (fields of the type `string`).

You could also format data in a script using the `formatter` ; see "Designer Script API" on page 874.

Date

Dates in variable data can be displayed as long, medium and short dates with different time displays. There are quite a few presets, but you can also enter a custom format mask.

1. Open the Text Script Wizard: double-click to open an existing script in the Scripts pane or create a new Text Script using the Text Script Wizard; see "Using the Text Script Wizard" on page 607.
2. Click a data field that contains text, or add such a data field to the script with the Add field button on the right.
3. Under **Format** you can choose one of the following options:
 - **Short Date** displays the day, month and year in two digits each, for example **01.04.16**.
 - **Medium Date** displays the day and month in two digits each and the year in four digits, for example **01.04.2016**. (This is also the value of the Default Date.)
 - **Long Date** displays the day as a number, the month's full name and the year in four digits, for example **1. April 2016**.
 - **Short Time** displays a time in hours and minutes in two digits each, for example **00:00**.
 - **Medium Time** displays a time in hours, minutes and seconds in two digits each, for example **00:00:00**. (This is also the value of the Default Time.)
 - **Long Time** displays a time in hours, minutes and seconds in two digits each, and adds a time zone, for example **00:00:00 EDT**.
 - **Short Date/Time** displays the date as a short date and the time as a short time, for example **01.04.16 00:00**.
 - **Medium Date/Time** displays the date as a medium date and the time as a medium time, for example **01.04.2016 00:00:00** (This is also the value of the Default Date/Time.)
 - **Long Date/Time** displays the date as a medium date and the time as a medium time, for example **1. April 2016 00:00:00 EDT**.

Alternatively, you can enter a **custom format mask**: click in the **Format** column for the corresponding field and start typing a **pattern** to format the date (and optionally, the time). Do not put the pattern in quotes. For possible patterns see "Date and time patterns" on page 918.

Note that this will only work if the type of the field has been set to Date in the Data Mapping Configuration and if the field contains a valid date.

Note

The locale influences the way dates, times, numbers and currencies are formatted; see "Locale" on page 590.

4. Close the Script Wizard. For a new script, don't forget to add the selector to the template.

Font style

Text originating from variable data can be displayed in uppercase, lowercase or proper case.

1. Open the Text Script Wizard: double-click to open an existing script in the Scripts pane or create a new Text Script using the Text Script Wizard; see "Using the Text Script Wizard" on page 607.
2. Click a data field that contains text, or add such a data field to the script with the Add field button on the right.
3. Under **Format** choose the correct setting:
 - **Uppercase** transforms all characters to uppercase.
 - **Lowercase** displays transforms all characters to lowercase.
 - **Propercase** transforms the first character of each word to uppercase and all other characters to lowercase.
 - **None** leaves the text as is.
4. Close the Script Wizard. For a new script, don't forget to add the selector to the template.

Numbers and currencies

Numbers, and strings existing of digits, can be displayed as a number with a certain formatting or as an amount of money. There are a few presets, but you can also type a format mask.

1. Open the Script Wizard: in the Scripts pane, double-click the script, or create a new Text Script using the Text Script Wizard; see "Using the Text Script Wizard" on page 607.
2. Click the data field that contains the numeric value that you want to display differently, or add the data field to the script with the Add field button on the right.

3. Under **Format** choose one of the following settings:
 - **Grouped** displays a number with three decimal places and sets the thousands separator for the value based on the current locale; see "Locale" on page 590.
 - **Currency** displays a number as an amount of money, with a thousands separator and rounded to two decimal places, based on the current locale; see "Locale" on page 590.
 - **Currency no symbol** does the same as Currency, but omits the currency symbol.
 - **Leading zero** adds a leading zero to a floating value between 0 and 1. This format is only available for fields that contain a `float` value. Note that when you open a data file or database without a Data Mapping Configuration, all fields are of the type `string`.
 - Σ (**Sum**) and $\Sigma\uparrow$ (**Sum Up**) are used in Dynamic Tables in a Print context. Σ is for transport rules at the end of a page and $\Sigma\uparrow$ shows the subtotal of the previous page.

Alternatively, you can enter a **custom format mask**: click in the **Format** column for the corresponding field and start typing a **pattern**. For example, the pattern `000000` means that the number should count six digits; leading zeros are added to numbers shorter than six digits. For an overview of pattern symbols see "Number patterns" on page 925 and <http://docs.oracle.com/javase/7/docs/api/java/text/DecimalFormat.html>. Note that for this to work, in the DataMapper the field that contains the value must be set to `SmallInteger`, `BigInteger`, `Float`, `SmallCurrency` or `LargeCurrency`.

4. Close the Script Wizard. For a new script, don't forget to add the selector to the template.

Showing content conditionally

One way to personalize content is to show or hide one or more elements depending on a field's value. For example, a paragraph written for Canadian customers could be hidden when the recipient of the letter is not living in Canada, if that can be derived from the data.

The Conditional Script Wizard helps you to show or hide one element – a paragraph, image or other HTML element - based on the value of one or more data fields. For example, you could check whether the data field 'State' is 'Equal To' the value 'British Columbia' or 'Québec', to include a paragraph for all recipients in those states.

Showing or hiding elements using the Conditional Content Script wizard

1. Right-click the element and click **Make Conditional**. Alternatively click the black triangle on the **New** button on the **Scripts** pane at the bottom left of the window, and click

Conditional Content Script. The Conditional Content Script wizard opens.

2. Rename the script so that it reflects what the script does.
3. If you have started creating the script from the **Scripts** pane, you have to type a **Selector**. The selector selects one or more pieces of text or elements from the template, so that the conditional content script can hide or show those pieces. An ID (for example: #conditional-script) is best if you want to show or hide one element only. Use a class selector (for example: .conditional) if the script should show or hide more than one element. See "Using the Text Script Wizard" on page 607 for further explanation on selectors.

If you have started the Conditional Script Wizard by right-clicking an element, you don't have to set a selector. If the element didn't have an ID, a new ID has been generated automatically. The new ID functions as the selector of the script.

You can change the selector after closing and reopening the script (double-click the name of the script in the **Scripts** pane).

4. Set the **Action**: use the drop-down to select whether to **Show** or **Hide** the element when the condition below is true.
5. Click the downward pointing arrow next to **Field**, to select the data field that should be evaluated.
6. Click the downward pointing arrow next to **Condition** to expand the list of conditions with which the data field can be evaluated. The options are: **Equal to**, **Not equal to**, **Contains**, **Does not contain**, **Begins with**, **Ends with**.
7. Type the **Value** or values (each on a new line) that should be used for the conditional check. Values are case sensitive. Dates should be entered in ISO standard notation (yyyy-mm-dd).

The selected action will be performed if the condition evaluates to true with one of the given values. If, conversely, the condition evaluates to **false**, and the option **Toggle Visibility** is checked, the opposite action will be performed. By default, this option is checked.

Note

If you need more complex conditions, click **Expand** and edit the code of the script. See "Writing your own scripts" on page 624.

8. Click **Apply** or **OK**.

9. To see the result, toggle to the **Preview** tab at the bottom of the workspace (or select **View > Preview View** on the menu).

Showing or hiding several elements with one conditional script

To apply one conditional content script to several elements, you have to use a CSS **class** or HTML element as the selector of the script. When using a CSS class, apply that class to the elements in question:

1. Double-click the conditional script in the **Scripts** pane to reopen it, or create a new conditional content script and follow the actions described in "Showing or hiding elements using the Conditional Content Script wizard" on page 613.
2. Change the selector to a CSS class (for example, `.male`) or to an HTML element with a certain CSS class (for example, `p.male`). See "Using the Text Script Wizard" on page 607 for further explanation on selectors.
3. Apply the same CSS class to all elements that should be shown or hidden under the condition that you have set in the conditional script. Click each element and type the class (without the preceding dot) in the **Class** field.

Showing or hiding a text selection

When you right-click on an element and make it conditional, the element as a whole will be made conditional. This happens even when you select a few words in a paragraph and right-click those words; the paragraph as a whole will be made conditional.

It is, however, possible to partially show or hide a paragraph or a line item in a list. Before you can do that, you have to select the text that you want to be shown or hidden and wrap it in a span element first:

1. Select the part of the text that you want to make conditional.
2. Right-click the selected text and click **Wrap in span**.
3. Type an **ID** and/or a **class**. An ID is fine if this is the only thing that should be shown or hidden on a given condition. Use a class if there is more that should be shown or hidden on the same condition.
4. Start creating a conditional content script from the **Scripts** pane. Use the ID or class as the selector of the script. See "Showing or hiding elements using the Conditional Content Script wizard" on page 613.

Conditional Print sections

You can include or exclude entire Print sections from the output, depending on a field's value. This can be done using the Conditional Print Section Script Wizard, described below. Alternatively you could write a Control Script (see "Control Scripts" on page 645).

Including or excluding Print sections using the Conditional Print Section Script wizard

1. Right-click the section and click **Make Conditional**.
Alternatively click the black triangle on the **New** button on the **Scripts** pane at the bottom left of the window, and click **Conditional Print Section Script**. Double-click the new script to open the Conditional Print Section Script wizard.
2. Rename the script so that it reflects what the script does.
3. Select the section you want to put a condition on.
4. Set the **Action: Print** or **Skip** that is performed when the condition below is **true**. The opposite action is applied when the condition returns false.
5. Click the downward pointing arrow next to **Datafield**, to select the data field that should be evaluated.
6. Click the downward pointing arrow next to **Condition** to expand the list of conditions with which the data field can be evaluated. The options are: **Equal to**, **Not equal to**, **Contains**, **Does not contain**, **Begins with**, **Ends with**.
7. Type the **Value** or values that should be used for the conditional check. Each additional value should go on a new line. The action is performed if the condition evaluates to true with one of the given values. Values are case sensitive. Dates should be entered in ISO standard notation (yyyy-mm-dd).

Example

You could check whether the `Province` field is 'Equal To' the value `Québec`, in order to print or skip a section only for customers living in Québec. With these values:

`Québec`

`Ontario`

the section will be printed if the `Province` field reads `Québec` OR `Ontario`.

Note

More complex conditions can be written in the Script Editor: click **Expand** and edit the code of the script. See "Control Scripts" on page 645.

8. Click **Apply** or **OK**.
9. To see the result, toggle to the **Preview** tab at the bottom of the workspace (or select **View > Preview View** on the menu). Take a look at the Resources pane: on each Print section that is affected by a Conditional Print Section script a small decorator appears, showing whether it will be skipped or printed with the current data record.

Dynamic Images

Dynamic images are called dynamic because they are switched, depending on the value of a data field. This way, a template can be adjusted to different customers.

Adding dynamic images

Dynamic images can be added to the template using the Dynamic Image Script Wizard only if you have:

- One or more data fields that contain values on the basis of which the images can be switched.
- An appropriate image for each group of customers. All files should be of the same type and they need to be stored in one folder (the **Images** folder on the **Resources** pane, or an external folder). It is important that they are named after the various possible values of the related data field. Adding dynamic images that are not named after a data field value requires a self-made script.

To use the Dynamic Image Script Wizard:

1. Add one image to the template. See "Adding images" on page 539.
2. Right-click the image and click **Dynamic Image**. Or select the image and click **Source** (not the field, but the label before the field) in the Attributes pane.
The Dynamic Image Script Wizard opens.
The image's ID is used as the script's selector. If the image did not have an ID, it is automatically generated.

The Dynamic Image Script Wizard composes a file name (including the path) based on the value of a data field, a prefix and a suffix:

- The prefix shows the path of the image.
- The suffix states the file extension of the image.
- The file name is the value of the data field(s) in the **Field** column.

The prefix and suffix are derived from the current image.

3. If necessary, enter another **Prefix** and/or **Suffix**.
4. Click the first field in the column **Field**, and then click the downward pointing arrow. Select the data field to be evaluated. Click the button **Add**, to add more fields if you want the file name to be composed of the value of several data fields. Note that only the suffix of the last data field should hold the file extension. The resulting file name, including the path and file extension, is assigned to the **src** (source) attribute of the image. You can click **Options** to verify this.
5. Click **Apply** or **OK**. Now click the **Preview** tab and browse through the records to verify that the script works as expected.

Tip

The dynamic images feature can be used to insert dynamic signatures, as described in this how-to: [Dynamic signatures](#).

How to insert dynamic images if there are no data fields with the actual names of the images is described in another how-to: [Dynamic image that doesn't contain the data field value](#).

Editing a Dynamic Image

To edit dynamic images added to the template earlier, right-click the image, or the space reserved for the dynamic images. Then click **Dynamic Image** to open the Dynamic Image Script Wizard again.

Dynamic table

In invoice templates, a Dynamic Table is an essential element. A Dynamic Table is different from a standard table in that it has a **variable** number of rows. In a Print context it will

automatically overflow into as many pages as necessary to output all rows and it can display a transport line.

Dynamic Tables are only available when the loaded record set or Data Mapping Configuration contains transactional data in one or more **detail tables** (see "Loading data" on page 594).

Creating a Dynamic Table

To create a Dynamic Table:

1. Open the Insert Detail Table dialog. There are several ways to do that:
 - Drag the data field that contains the name of the detail table into the template.
 - On the menu select **Insert > Table > Detail**.
 - On the toolbar, click the **Insert detail table** button.
2. Enter the table's desired attributes:
 - **ID**: A unique identifier for the table. IDs are used to access the table from scripts and as CSS selectors for style rules.
 - **Class**: A class identifier for the table. Classes can be shared between elements and are used to access the table from scripts and as CSS selectors for style rules.
 - **Detail Table**: Use the drop-down to select which detail table to display within the dynamic table.
 - **Width**: Enter the width of the table.

A Dynamic Table is always inserted at the cursor position.

3. Click **Next** and select which fields should show up in the Dynamic Table. The order of the fields indicates in which order columns are displayed in the dynamic table, from left to right. Select a line and then use the **Up** and **Down** buttons to change the order of the columns. You could change the placeholder for each data field as well; just click a placeholder to edit it.
4. Click **Next** and check **Calculate Subtotals** to enable the options for a (sub)total at the end and (in Print sections) transport lines. The options are:
 - **Column**: Use the drop-down list to select the field that contains the currency value to be used to calculate the subtotal of the table. This field generally contains the result of item prices multiplied by the quantity.

- **Field name:** Type the name to display in the footer when displaying that page's subtotal.
 - **Show in footer:** Check to display the subtotal in the footer of the table at the bottom of each page **and** at the end of the table.
 - **Show in header** (transportline): Check to display the subtotal of the previous page at the top of the table.
5. Click **Next** and select the styling attributes. Use the drop-down to select the desired table style. Choose **No Style** if you want to style the table yourself.
 6. Check the option **Hide when empty** to make the table invisible when there are no data to display in it.
 7. Click **Finish** to add the table to the section.

When a Dynamic Table is added, a script is created for each of the columns containing a placeholder for a field that is to be replaced. These scripts are placed inside a folder named after the table's ID, on the **Scripts** pane.

Note

Using nested detail tables in the **Designer** module requires scripting, as described in this How-to: [Cloning your way through nested tables](#).

Adding a row at the bottom or the top of a Dynamic Table

Sometimes you'll want to add one or more rows to the header or footer of a Dynamic Table: to add taxes and/or the total of the invoice to the table, for example, or to add a custom message. A header or footer row can be added to a Dynamic Table as follows:

1. In the workspace, open the **Design** tab. Right-click the first line of the table if you want to add a header row, or the last line if you want to add a footer row.
2. On the shortcut menu select **Row > Insert below** or **Insert above**. The new row will be added to either the header or footer.
3. Right-click the row and choose **Row > Show**. Now you have the following options:
 - A header row marked as a **Transport line** will not appear at the very top of the table, but only on following pages if the table gets split over multiple pages. Note that only the first row that is marked as Transport line will be taken into account. To

make a header row appear at the start of the table and on following pages, make sure that it is not marked as Transport line.

- A footer row can appear before each page break (**Before page break**), if the table gets split over multiple pages, or only at the end of the table (**At end of table**), or before each page break *and* at the end (**Always**).

You can fill additional rows as usual. You could for example drag a data field to the new row (see "Variable Data" on page 604) or type in the cells.

Examples

For a few examples of how to adjust the default subtotals footer and (transport line) header, see the following how-to: [Custom table overflow footers](#).

Styling a Dynamic Table

The Insert Detail Table wizard lets you select a style, but if you want to apply a different style to the table, choose **No Style** when creating the table. Then the style of a Dynamic Table is completely customizable: you can change the font, font size and color, the borders, the cell padding (the distance between the edge of the cell and its content), and the background color or image of the table and its cells. See "Styling a table" on page 571.

Note

When generating output from a template, a Dynamic Table is created slightly faster when it's styled via Cascading Style Sheets than when it's styled with local formatting. Therefore the preferred way to style a dynamic table is via style sheets.

Change detail line formatting based upon a data field value

An example of how to change the formatting of a line in a Dynamic Table, based upon a data field value, is given in the following how-to: [Change detail line formatting based upon a data field value](#).

Resizing a Dynamic Table

To change the width of a Dynamic Table or of a column in a Dynamic Table, select it (see "Selecting an element" on page 469) and type the desired width as a percentage in the respective field on the **Attributes** pane.

The height of the Dynamic Table is adjusted automatically to the amount of data added to it in Preview mode or when generating output.

It is however possible to change the height of the rows: click in the row and type the desired height in the respective field on the **Attributes** pane. All line item rows will have the same height.

Hiding an empty Dynamic Table

The number of rows in a Dynamic Table is variable, as it depends on a detail table in the data. You might want the Dynamic Table to be hidden when there are no data to display. There are two ways to achieve that.

- When creating a Dynamic Table, you can check the option **Hide when empty**. (See "Creating a Dynamic Table" on page 619.)
- For an existing Dynamic Table you can check the option **Hide when empty** on the **Attributes** pane.

HTML elements and attributes

In HTML, a Detail Table is just a normal `<table>` element with rows and cells (see "HTML element: table" on page 543). But apart from the native attributes of a table, row and cell element, some `data-` attributes can be seen in detail tables:

- `data-detail`: The name of the detail table in the data, for example: `data-detail="products"`.
- `data-repeat`: The row will be repeated if it has this attribute: `data-repeat=""`.
- `data-showin`: This attribute determines the visibility of the row in different situations, if the table gets split over multiple pages:
 - `header` will make the row show up at the top of the table on the first page only.
 - `footer` will make the row show up in the footer of the table on the last page only.
 - `break` used in a row in the `<thead>` section of a table indicates that the row should not be displayed at the top of the table on the first page, but only on following pages. Used on a row in the `<tfoot>` section, it indicates that the row should be displayed before each page-break. This value may be combined with `footer` or `header`, for example: `data-showin="footer, break"`, to make the row show up on every page.

Note that these options can also be set via the user interface: right-click on the row and

select **Row > Show**; see "Adding a row at the bottom or the top of a Dynamic Table" on page 620.

- `data-breakable`: this attribute is added to every copied row (in preview mode or when creating output), in each of them with a unique ID as its value. This is required by the pagination routines of Connect to split the table across pages.
- `data-column-resize`, if present, indicates that the columns may be resized (`data-column-resize=""`).

Personalized URL

Personalized URLs (pURLs) are links that are tailor-made for a specific purpose, generally for individual clients. They can serve multiple purposes, for instance:

- **Click Tracking**: A unique ID in the link makes it possible to track the source of the click (for example, a link in an email campaign).
- **User Tracking**: A user-specific ID reveals who clicked the link and at what time.
- **Landing Pages**: Information in the link invokes a unique landing page with specific products or services.
- **Personalized User Pages**: Using information from a database, a user is served a completely personalized web page with their name and information tailored to them, enhancing user response.

Typically, a pURL in a Connect template takes the user to a personalized landing page, for example, to download an invoice or get access to specific products or services.

In addition to the pURL, to generate a personalized landing page the Connect Server needs a template with a Web context and a Workflow process with the following tasks:

- A HTTP Server Input task to capture incoming web requests (see Workflow Help: [HTTP Server Input](#)).
- An Execute Data Mapping task to create the record set appropriate for the template (see Workflow Help: [Execute DataMapping Task](#)).
- A Create Web Content task that generates the HTML files (see Workflow Help: [Create Web Content](#)).

Creating a personalized URL

Creating a personalized URL implies writing a script. See "Writing your own scripts" below.

It also requires some planning, because the pURL needs to contain data that is necessary to create the web page. For instance, creating a personalized URL for a client's invoice may require the Invoice Number to be present in the URL, which is then used to retrieve the invoice data, generate the invoice in PDF or HTML format using a template, and then return it to the browser. The trick is then to add the designated information to a hyperlink. How to do this is described in a how-to; see [How to dynamically insert a hyperlink](#).

Writing your own scripts

Personalization can be taken a lot further than just inserting names and addresses, and hiding or showing text or images. Every bit of information in your communications can be made entirely personal, using scripts.

A script is a small set of instructions to the program, written in JavaScript. When Connect generates the actual output – letters, web pages or emails -, it opens a record set and merges it with the template. It takes each record, one by one, and runs all scripts for it (in a specific order, see "The script flow: when scripts run" on page 660).

This topic explains how scripts work and how you can create and write a script. Most scripts can be made using one of the Script Wizards. For a block of variable data, such as an address, the Text Script Wizard is a perfect fit. Paragraphs can be made conditional with a Conditional Script Wizard. For dynamic images, you can use the Dynamic Image Script Wizard. In an Email context, you are provided with a number of Script Wizards to set the sender, the recipients and the subject of the email. However, when you want to do something that goes beyond what you can do with a Wizard, like creating a conditional paragraph with a condition that is based on a combination of data fields, you have to write the script yourself.

Script types

There are generally two types of scripts in the Designer: **Control Scripts** and **template scripts**.

Control Scripts don't touch the content of the sections themselves, but they change the way a template is outputted, for example by selecting or omitting sections from the output. For more information about Control Scripts and their use, see "Control Scripts" on page 645.

Template scripts can change the contents of sections in a template. This type of script must have a **selector**. The selector can be text, an HTML element and/or a CSS selector (see

"Selectors in Connect" on page 660). Running a template script starts with looking for pieces of content in the template that match the script's selector.

The results of this query can vary from one occurrence of a simple text (for example: @EMAIL@) to a large collection of HTML elements. For example, when the selector is **p**, the HTML tag for a paragraph, all paragraphs will be collected and passed to the script.

Tip

Hover over the name of a script in the **Scripts** pane to highlight parts of the template that are affected by the script.

Next, the script can modify the selected pieces of content, using values from the record that is merged to the template at the time the script runs. It can, for example, hide, replace or add text or change the style of those pieces of content. This is how scripts personalize documents.

Note

In a Print context, the scripts in the Scripts pane run once for each section and then once for each Master Page (see "Master Pages" on page 350).

Creating a new script

Writing a template script starts with this procedure:

1. On the **Scripts** pane at the bottom left, click **New**. A new script appears in the list. Double-click on it to open it.
2. Change the name of the script, so that it reflects what the script does.
3. Choose which kind of **selector** you want to use. Running a script starts with searching the template for pieces of content that match the script's selector. The collected pieces of content are passed on to the script, so that the script can modify them.

The selector can be:

- **Text**, for example: @lastname@, or {sender}. The text doesn't have to have any special characters, but special characters do make it easier to recognize the text for yourself. In the Script Wizard, click **Text** and type the text to find.

- A **selector** (HTML/CSS):
 - HTML elements of a certain type, such as a paragraph: <p>. In the Script Wizard, click **Selector** and type the HTML tag in the Selector field without the angle brackets: p.
 - HTML elements with a specific CSS class (eg. green). In the Script Wizard, click **Selector** and type the class name in the Selector field , preceded by a dot: .green.
 - An HTML element with a specific ID (eg. intro). In the Script Wizard, click **Selector** and type the ID in the Selector field , preceded by #: #intro. In an HTML file, each ID should be unique. This means that a particular ID can be used only once in each section.
 - Etcetera. See http://www.w3schools.com/cssref/css_selectors.asp for more selectors and combinations of selectors; also see "Selectors in Connect" on page 660 for selectors that can only be used in Connect.
- A **selector and text**. This is text inside an HTML element (or several HTML elements) with a specific HTML tag, CSS class or ID. In the Script Wizard, click **Selector and Text**.

Tip

When output speed matters, choose **selector** or **selector and text**. Searching text is a rather lengthy operation, compared to searching for HTML elements and/or CSS selectors. See also: "Testing scripts" on page 632.

There is a shorter route to create a script for an element with a specific ID:

1. In the template, click the element for which you want to create a script.
2. On the **Attributes** pane at the top right, type an ID. (In HTML, IDs start with #, but in this field you should type it without the preceding #).
3. Click the label to the left of the ID input field (ID) to make a new script with the ID that you typed as a selector.

Writing a script

1. Create a new script (see: "Creating a new script" on page 625), or double-click an existing script in the **Scripts** pane on the bottom left.
If the script was made with a Script Wizard, you have to click the **Expand** button before you can start writing code. This will change the Script Wizard into an editor window.

Warning

When you change an expanded text script and save it, it becomes impossible to edit the script using the Script Wizard again.

2. Write the script. Click **Apply** from time to time to see if the script works as expected. This will be visible on the **Preview** tab in the main workspace.

Syntax rules

Every script in the Designer must follow JavaScript syntax rules. For example, each statement should end with ; and the keywords that can be used, such as **var** to declare a variable, are JavaScript keywords. There are countless tutorials available on the Internet to familiarize yourself with the JavaScript syntax.

For a simple script all that you need to know can be found on the following web pages: http://www.w3schools.com/js/js_syntax.asp and http://www.w3schools.com/js/js_if_else.asp.

A few examples can be found in a How-to: [Combining record based conditions](#).

Tip

In the editor window, press **Ctrl + Space** to see the available features and their descriptions.

Use the arrow keys to select a function or object and press Enter to insert it in the script.

Type a **dot** after the name of the function or object and press Ctrl + space again to see which features are subsequently available.

For more keyboard shortcuts, see "Keyboard shortcuts" on page 738.

Two basic code examples

Writing a script generally comes down to modifying the piece(s) of content collected from the template with the script's selector, using values, or depending on values of the record that is being merged to the template at the moment the script runs.

Modifying the template

To access and change the results of the query that is carried out with the selector (in other words: to modify the output), use the object **results**.

The following script (with the selector **p**) changes the text color of all paragraphs to red with a single line of code:

```
results.css('color', 'red')
```

It does this for each and every customer, because it does not depend on a value from the record that is being merged to the template.

Using values from the record in a script

To access the record that is being merged to the template when the script runs, use the object **record**.

Suppose you want to display negative amounts in red and positive amounts in green.

Assuming that there is an AMOUNT field in your customer data, you could write the following script (with the selector: **td.amount**, that is: table cells with the class 'amount').

```
var amount = record.fields.AMOUNT;
if (amount >= 0)
    {results.css('color', 'green');}
else if (amount < 0) {
    results.css('color', 'red');
}
```

When this script executes, it stores the value of the AMOUNT field from the current record in a variable and evaluates it. If the value is zero or higher, the color of text in the **results** - the table cells in this case - will be set to green; if the value is below zero, the text color will be set to red.

Tip

For more examples of using conditions, see this how-to: [Combining record-based conditions](#).

Designer API

Features like **results** and **record** do not exist in the native JavaScript library. These are additional JavaScript features, designed for use in Connect scripts only. All features designed for use in the Designer are listed in the Designer's API, with a lot of examples; see "Designer Script API" on page 874.

Managing scripts

Changing the order of execution

When a record set is merged with a template to generate output, all scripts are executed once for every record in the record set, in the order in which they appear in the **Scripts** pane at the bottom left.

The order in which scripts are executed is particularly important when one script produces content that contains a selector for another script. If the other script has already been executed, it will not run again automatically. So, scripts that produce content that contains one or more selectors for other scripts, need to come first.

To change the order in which scripts are executed:

- Click a script or a folder in the **Scripts** pane at the bottom. Drag it up or down and drop it.

Note

Control scripts are always executed first, regardless of where they are in the Scripts pane. They can not be excluded from execution for a specific context or section, using the execution scope of a folder; see "Execution scope" on the facing page. What you can do is disable the script or the containing folder; see "Enable/disable scripts" on page 631.

Script folders

Scripts can be organized in folders. Why would you do that? For three reasons:

- Folders have an execution scope. You can specify for which contexts and sections the scripts in a folder have to run.
- Folders provide a better overview than a long unorganized list of scripts.

- Folders make it easier to change the order of execution for a bunch of scripts (see: "Changing the order of execution" on the previous page to learn why the order of execution is important). Dragging a folder up or down will cause all the scripts in that folder to be executed earlier or later, respectively.

To make a new folder on the **Scripts** pane:

1. In the **Scripts** pane, click the black triangle on the **New** button.
2. Click **Folder**. The folder will appear in the list of scripts.
3. Change the name of the new folder: right-click the folder and click **Rename**.
4. Drag scripts to the folder.

Tip

It may be helpful to put scripts that have an effect on the same context or section in one folder, because you can set the execution scope of scripts per folder (see: "Execution scope" below).

Note

Control scripts are always executed first, regardless of where they are in the Scripts pane. They can not be excluded from execution for a specific context or section, using the execution scope of a folder; see "Execution scope" below. What you can do is disable the script or the containing folder; see "Enable/disable scripts" on the next page.

Execution scope

A particular script may be used in one context or section, but not in other contexts or sections. Nevertheless, when processing the template, the Designer tries to find the selector of each script in all contexts and sections – unless the script is located in a scripts folder for which the execution scope has been set to the relevant contexts or sections. So, setting the execution scope of a folder saves processing time.

To change the execution scope of a script:

1. Put the script in a folder; see "Script folders" on page 629.
2. Right-click the folder, and then click **Properties**.
3. Check the contexts and sections for which the scripts in this folder should run.

Note

Control scripts are always executed first, regardless of where they are in the Scripts pane. They can not be excluded from execution for a specific context or section, using the execution scope of a folder; see "Execution scope" on the previous page. What you can do is disable the script or the containing folder; see "Enable/disable scripts" below.

Tip

For more ways to optimize scripts, see "Optimizing scripts" on page 636.

Enable/disable scripts

A disabled script will not run at all when the template is merged with a record set to generate output. Disabling script execution in certain contexts or sections helps with performance, since scripts normally run, whether or not their placeholder or selector is present in your template. It is highly recommended to disable any script that is not relevant to specific sections or contexts.

When you disable a folder, all scripts in the folder will be disabled.

To enable or disable a script or a folder:

- On the **Scripts** pane, right-click the script or the folder and click **Disable** (if the script or folder was enabled) or **Enable** (if the script or folder was disabled).

Tip

For more ways to optimize scripts, see "Optimizing scripts" on page 636.

Import/export scripts

Scripts can be exported - one at a time - for use in other templates. To do this:

1. On the **Scripts** pane, click on a script, and then click the **Export** button, or right-click a script and select **Export**.
2. Give the script a name and click **OK**.

To import a script in a template:

- On the **Scripts** pane, click the **Import** button. Find the script and click **OK**

Files that a script may refer to, such as images, snippets and fonts, are not exported or imported together with a script.

Test the script to make sure that all files are present in the template and that the script's selector matches something in the content of the template; see "Testing scripts" below.

Testing scripts

The quickest way to test that scripts work as expected, is to click the **Preview** tab at the bottom of the workspace.

You can even do this while creating a new script, either with a Script Wizard or in the expanded script editor. Click **Apply** at the bottom of the script editor to see the effect of the script on the **Preview** tab of the Designer.

Note that scripts that use values of data fields can only be effective when a data file or Data Mapping Configuration is open. See "Loading data" on page 594.







Testing for errors




One way to see if a script is functional is to take a look at the Scripts pane.

Tip

Hover over the name of a script in the **Scripts** pane to highlight parts of the template that are affected by the script.

Icons on the name of scripts in the **Scripts** pane can show a warning, information or error icon.

 Spa Location	#spa-location
 Promo	#promo1
 Date	@Date@
 Year	@Year@
  Address	
 Employee	@EMP@

- The information icon  (i) shows that the selector of the script does not produce a result in the current section.
- The warning icon  (!) appears, for example, when a script refers to an unknown field in the record set, or when ; is missing after a statement.
- The error icon  (x) displays when the script results in an error, for example, when it uses an undeclared variable.

Preflight

In addition to the icons and messages in the Scripts pane, something else can show if your scripts function as expected before generating output:

1. On the menu, select **Context > Preflight**.
2. Select **All**, or enter a selection of records. You can specify individual records separated by semi-colons (;) or ranges using dashes. For example: 2;4;6-10 would print pages 2, 4, 6, 7, 8, 9 and 10.
3. Click **OK**.

Preflight executes the template without actually producing output. When a data mapping configuration is used, any pre- and postprocessors are run as well.

The Preflight window displays any issues once it's done. It will tell, for example, which selectors were not encountered in the template.

Double-click a script warning/error (either in the Preflight Progress dialog or in the Preflight Result view) to open the script in the script editor. The relevant line will be highlighted.

Tip

Be aware that scripts run in a specific order (see "The script flow: when scripts run" on page 660). When one script unintentionally influences the results of another script, changing the order of the scripts in the Scripts pane may help (see "Changing the order of execution" on page 629).

Testing for speed issues

To measure the time that the execution of scripts will take:

- On the **Context** menu, click **Profile scripts**.

Profiling means running the scripts in the template, with the current record, to see how fast scripts in the **Scripts** pane execute. It helps greatly in troubleshooting performance issues caused by scripts.

After running the Script Profiler you can see in which sections the script has run:

- Hover the mouse over a value in the column **Count** to see the number of times that the script has run, per section.

You can also see the breakdown of the execution time across different execution stages:

- Hover the mouse over a value in the column **Elapsed** to see the time elapsed (in milliseconds) since the start of the session. In the Scripts Profiler, the scripts are by default sorted based on the values in the **Elapsed** column, from high to low.
- Hover the mouse over a value the column **Delta** to see the difference between the time elapsed (in milliseconds) in the previous session and in the current session.

The script execution stages are:

Query: the time it takes to find the selector in the template.

Tip

Looking for text is a rather lengthy operation. Use an ID (possibly in combination with a text) instead of a text selector to make the query faster. For more tips, see "Optimizing scripts" on page 636.

Execution: the time it takes to execute the script. If you are an experienced JavaScript coder you may be able to optimize the code to speed up the execution of the script.

Tip

Functions that actually change the content of the template (for example, **append()**) are comparatively time consuming. Avoid using such functions in a loop. For more tips, see "Optimizing scripts" on the facing page.

Note that the times vary slightly per run of the Script Profiler. Run the Script Profiler a number of times and calculate an average from the results, before trying to speed up the execution of a script.

Script Profiler settings

Number of runs

By default, the Script Profiler runs on 1000 instances of all the scripts. To test on a higher or lower number of instances:

1. On the menu, select **Window > Preferences**.
2. Click **Scripting**.
3. Set a number of iterations (maximum one billion) and click OK.

Sorting

In the Scripts Profiler, the scripts are by default sorted based on the values in the **Elapsed** column, from high to low. Click any of the columns to sort the scripts according to the values in that column.

Script timeout

When testing scripts, either by toggling to Preview mode or by using the Script Profiler, a script timeout is active in the Designer, so that scripts that need a very long time to run are stopped after a set time. You can adapt this timeout to your needs, as follows:

1. On the menu, select **Window > Preferences**.
2. Click **Scripting**.
3. Set a timeout in seconds (for example: 2s) and click OK. The minimum timeout is 1 second.

Note

The script timeout is not active when generating output.

Optimizing scripts

In the process of output generation, the execution of scripts may take up more time than necessary. To optimize a template, it helps to disable scripts that don't have an effect on the output; see "Managing scripts" on page 629.

This topic presents a number of other ways to speed up script execution by optimizing the scripts.

Use an ID as selector

Scripts (except Control Scripts) start with a query. The **selector** in the second column in the **Scripts** pane is what a script looks for in the template. If you've used the drag-and-drop method (without pressing the Alt or Ctrl key) to insert a data field in a template, the selector is a small text: the name of the data field surrounded by @ signs, @firstname@ for example.

Looking for text in a text is a less optimized operation and may impact output speeds in longer documents. To speed up the output process, point the script to the element that contains the placeholder, by using its ID as selector. This narrows the scope of the search and results in a very fast query, as elements with an ID are indexed by Connect Designer's layout engine.

To learn how to put a placeholder or placeholders inside an element that has an ID, see "Boxes" on page 513. To use that ID as the script's selector: double-click the script in the Scripts pane and change the Find method to **Selector and Text**, or to **Selector** if the placeholder is the only content of the container. Enter the ID of the wrapper element in the **Selector** field, preceded by #, for example: #firstname.

Tip

When using the drag-and-drop method to insert data fields in a template:

- Press the **Alt** key while dragging, to wrap the placeholder in a **span**, give the span an **ID** and have that ID used as the script's selector.

- Press the **Ctrl** key while dragging, to wrap the placeholder in an absolute positioned box (a **div**) at the cursor position. A unique ID is assigned to the box and used as the script's selector.

Avoid DOM manipulations

The Scripting API of the Designer is a very powerful tool to manipulate and personalize your document. But keep in mind that DOM manipulation commands like `append()`, `prepend()`, `before()` and `after()` are resource intensive.

Try avoiding DOM modifications, especially within loops. Storing the content in a variable and appending the information after the loop is more efficient: this way, the template will be touched only once.

Example

The following example loads a snippet into a variable and uses the `find()` and `text()` commands of the Designer scripting API.

```
var labelElm = loadhtml('snippets/label.html');
for(var i = 0; i < record.tables.products.length; i++) {
    var label = labelElm.clone();
    label.find('@ProductLabel@').text(record.tables.products
[i].ProductDescription);
    results.after(label);
}
```

What's wrong with this code is that it inserts the personalized information **within** the loop. The `after()` command runs as many times as there are records in the detail table 'products'.

The script below is much more efficient: it adds the personalized content to a string called `labelStr` and only calls `after()` after the `for` loop.

```
var labelElm = loadhtml('snippets/label.html');
var labelStr = "";
for( var i = 0; i < record.tables.products.length); i++) {
    var label = labelElm.clone();
    label.find('@ProductLabel@').text(record.tables.products
[i].ProductDescription);
    labelStr += label;
```

```
}
results.after(labelStr);
```

Use replace()

When personalizing HTML fragments retrieved from a snippet or from the template itself, JavaScript's `replace()` method shows the best performance.

`Replace()` can only be used on Strings, while the commands `loadhtml()` and `query()` return or a `QueryResult`, which is a set of strings, like the `results` object.

A `QueryResult` allows you to perform DOM manipulations like adding and removing elements, adding and removing CSS classes etc. When the required manipulations are limited to find/replace actions, you could change the `QueryResult` into a string. This allows you to replace text using the `replace()` method.

For this, you could use `toString()`:

```
var labelSnippet = loadhtml('snippets/label.html').toString();
```

Or you could copy the HTML of the `QueryResults` to a variable:

```
var block = results.html();
```

Example

```
var labelSnippet = loadhtml('snippets/label.html').toString();
var labelStr = "";
for( var i = 0; i < record.tables.detail.length; i++) {
    var label = labelSnippet;
    label = label.replace('#', i);
    label = label.replace('@product@', record.tables.detail[i].fields
['product']);
    label = label.replace('@notes@', record.tables.detail[i].fields
['notes']);
    label = label.replace('@netweight@', record.tables.detail
[i].fields['netweight']);
    labelStr += label;
}
results.after(labelStr);
```

Tip

The `replace()` method as used in the above example replaces only the first occurrence of the search string. To **replace every occurrence** of a search string in a given string, use a **regular expression**. In the following line of code, the regular expression `/@product@/g` makes `replace()` search for all occurrences of the string `@product@` in the `label` string:

```
label = label.replace(/@product@/g, record.tables.detail  
[i].fields['product']);
```

In this example, `@product@` is a pattern (to be used in a search) and `g` is a modifier (to find all matches rather than stopping after the first match). For more information about possible regular expressions, see http://www.w3schools.com/js/js_regexp.asp.

The script in this how-to: [Translate and replace script](#) uses the `replace()` method with a regular expression.

Replace several placeholders in one script

Suppose there are 20 different placeholders in a postcard (for the address, account and customer details, a promo code, the due date, discounts, a link to a personalized landing page etc.). Typically this would require 20 queries. Even after optimizing these scripts by using an ID as selector for those scripts, there are still 20 scripts, 20 queries to run.

If there was only one query, one single script to do all the work, the output could be generated much faster. Reducing the number of scripts improves the performance of the template. How to do this?

First, wrap the content that contains all of the placeholders in one (inline) `Box` and give that `Box` or `Span` an ID (on the `Attributes` pane). Next, create a script that uses that ID as selector. Then replace all placeholders in the script and put the content back in the template.

This is similar to working with snippets, but in this case the element is extracted from the actual template.

Example

The following script replaces all of the placeholders on a postcard. It takes advantage of the JavaScript `replace()` command. Assuming that the ID of the block that requires personalization is `promoblock`, the script has to have its selector set to `#promoblock`.

```
var block = results.html();
var data = record.fields;
block = block.replace('@name@',data.first + ' ' + data.last);
block = block.replace('@address@',data.address);
block = block.replace('@zip@',data.zip);
block = block.replace('@city@',data.city);
block = block.replace('@country@',data.country);
block = block.replace('@saldo@',data.saldo);
block = block.replace('@promo@',data.promo);
block = block.replace('@customercode@', data.customercode);
...
results.html(block);
```

The first line retrieves the HTML of the promo block and stores it in a variable called `block`. To make the code more readable, the fields from the record are stored in a variable named `data`. After replacing the placeholders by values, the script replaces the HTML of the promoblock with the personalized string.

Other resources

There are also many resources online to help learn about JavaScript performance and coding mistakes. See for example:

- [JavaScript performance](#)
- [The 10 most common JavaScript mistakes](#)
- [Tips for writing efficient JavaScript.](#)

Note that most resources on the web are about JavaScript in the *browser*, but the greatest majority of the tips do, indeed, apply to scripts in general, wherever they are used.

Loading a snippet via a script

Instead of dragging it into the content directly, it is possible, and often very useful, to load a snippet dynamically. Create a script (see "Writing your own scripts" on page 624) and in the code use the following function:

```
results.loadhtml('snippets/nameofthesnippet.html')
```

Remote snippets are retrieved in the same way, except that the file extension should be `.rhtml` instead of `.html`.

Note that the name of the snippet must be exactly the same as in the Snippets folder.

This function will insert the snippet in the content at any position where the script's selector is encountered.

For more examples, see "loadhtml()" on page 920.

Note

Make sure that the file name is exactly the same as the file in the **Snippets** folder. If the file name isn't correct, the snippet will not appear in the template.

Loading part of a snippet

When a snippet contains a part that can be identified by a selector, that selector can be used to load that part of the snippet into a template.

In script, use the following code:

```
results.loadhtml('snippets/nameofthesnippet.html', 'selector')
```

See "loadhtml()" on page 920 for more information about this function.

Loading a snippet, depending on the value of a data field

To load a snippet depending on the value of a data field, you have to add a condition to the script.

Example

The following script evaluates if the value of the LANGUAGE field in the record is 'En'. If so, the snippet is added to the content.

```
if (record.fields.LANGUAGE == 'En') {  
results.loadhtml('snippets/nameofthesnippet.html');  
}
```

Another example is given in a how-to; see [Load a snippet based on a data field value](#).

Loading part of a snippet, based on the value of a data field

When a snippet contains a part that can be identified by a selector, that selector can be used to load that part of the snippet into a template. It is possible to do this, based on the value of the data field. This is easiest when the selector matches the value of a data field.

Example

The following script reads the value of the LANGUAGE field in the record and uses that value as the selector in the function loadhtml(). If the snippet contains an HTML element with this ID (for example, <p ID="En">), that HTML element will be added to the content:

```
var language = record.fields.LANGUAGE;  
results.loadhtml('snippets/nameofthesnippet.html', '#' + language)
```

Another example is given in the following how-to: [Using a selector to load part of a snippet](#).

See also: "Designer Script API" on page 874.

Tip

An easy way to group content in a snippet is putting each part in a container and giving that container an ID, for example:

```
<div ID="EN"><p>This is text for English customers.</p></div>
```

Use the function `.children()` to load the contents of the container, and not the container itself. For example:

```
results.loadhtml('Snippets/myfooter.html', '#EN').children()
```

This script loads the paragraph of the example (<p>), but not the container itself (<div>).

Load a snippet and insert variable data into it

The following script loads part of a snippet based on the value of a field, and then finds/replaces text by the value of a field before inserting the content into the document.

```
var promoTxt = loadhtml('snippets/promo-en.html', '#' +  
record.fields['YOGA']);  
promoTxt.find('@first@').text(record.fields['FIRSTNAME']);  
results.html(promoTxt);
```

Loading content using a server's RESTful API

Content in a template is usually static (apart from being personalized) and part of the main text flow. It can also be located in a snippet (see "Snippets" on page 548).

It is also possible to include content that is served by another server. Many servers provide an API to fetch publicly available content from their site. That content may even be dynamic: the **most recent** blog posts on a WordPress website, for example, or the **current** weather forecast for a certain city.

This topic explains how to retrieve content using a server's API and insert that content in a template.

Step 1: Getting the appropriate link

To request content from another server, you will need a link.

Some websites give the option to **embed** their content in your website by providing a link or the complete HTML. Youtube.com, for example, offers not only a link to share a certain video, but also the full HTML to embed that video in your website.

If that option is not available, you will have to build the link yourself. Find the server's RESTful **API** and look through it to get the exact endpoint and parameters that you need.

With many servers it is required to use an **API key** in the link; this key generally comes for free after you sign up to their website. The key will be part of the link that is used to make a request to the server.

Note

Pay attention to the service's Terms of Service. Many servers have limitations on the number of calls that can be made to them for free. Beyond these limits, their content will not show up in your template unless you purchase a business plan.

Step 2: Preparing the template

The next step is to set up a template. If you've got the HTML to embed content in your template you can paste that HTML on the Source tab (and skip Step 3).

Otherwise your template has to contain an element that can be replaced or followed by the remote content: an empty paragraph, for example, or a heading. If the element isn't unique in the template, give the element an ID.

Note that interactive content, such as an interactive map, can only be used in Web templates, and cannot be output on Print or Email contexts (even though they will show up in Preview mode!).

Step 3: Writing a script

The final step is to write a script that retrieves the content and inserts it into the template (see "Writing your own scripts" on page 624). Use the element or the ID of the element that you added in Step 2 as the script's selector. For information about selectors, see "Selectors in Connect" on page 660.

Tip

Select an element, then click on 'ID' in the Attributes pane, to create a script that has that element's ID as selector.

Retrieving content

Depending on the type of content that the remote server returns - HTML or JSON - you can use `loadhtml(location)` or `loadjson(location)` (see also: "loadhtml()" on page 920 and "loadjson()" on page 922) to retrieve the content. The link that you selected in Step 1 should be passed to the function as a string. For example:

```
loadjson('https://blog.mozilla.org/wp-json/wp/v2/posts?per_page=5');
```

If the returned content is JSON data, that data may have to be wrapped in HTML before inserting it into the template. This is demonstrated in the example below.

Tip

Install the Postman application to preview JSON returned by an endpoint.

Inserting content in the template

To insert the content after the selected element, use `results.after()`. To replace the element with the new content, use `results.html()` or `results.replaceWith()`.

Example: recent posts

The following script loads five posts from Mozilla's blog and inserts their titles as a list of links in a template. Mozilla's blog is a WordPress website. Since the WordPress REST API uses JSON as the response format, the `loadjson()` function has to be used.

If the script's selector was `h1` (a level one heading), the retrieved content would be inserted after each level one heading.

```
var postsObj = loadjson('https://blog.mozilla.org/wp-  
json/wp/v2/posts?per_page=5');  
var html = '';  
html = '<ul>';  
for (var idx in postsObj) {  
    html += '<li><a href="' + postsObj[idx].link + '">' + postsObj  
[idx].title.rendered + '</a></li>';  
}  
html += '</ul>';  
results.after(html);
```

See [WordPress REST API developer endpoint reference](#).

Tip

More examples of how to use a RESTful API to load external content are given in these How-to's:

- [Using the Google Maps API](#)
- [Using the OpenWeatherMap API](#)

Control Scripts

When output is generated from a template, Control Scripts run **before** all other scripts, when a record is merged with a context. They determine how different sections of the context are handled. They can, for example, make the page numbering continue over all Print sections, split Email attachments, or omit Print sections from the output.

Some knowledge of JavaScript is needed to edit Control Scripts, just as for any other self-made scripts, because there is no Control Script Wizard; see "Writing your own scripts" on page 624.

This topic explains how to add a Control Script and it gives an overview of what Control Scripts can do. It will also tell you where you will find information about each feature, including examples.

What Control Scripts are

Control Scripts are a special kind of Designer script. They can manipulate the way output is generated from a template. They allow you, for example, to change the page numbering in Print output, to split one generated Print document into multiple Email attachments, or to set a Print section's background dynamically. (These are only a few examples; for more uses of Control Scripts see "What to use a Control Script for" on the next page.)

Control Scripts differ from template scripts in two ways:

- Control Scripts run before all other scripts. When a template consists of several contexts, and these contexts are combined in the output - for example, when an Email is generated with the Print context as attachment - all scripts run once before each context; Control Scripts first.
- Control Scripts don't have a selector, like the other scripts do. A selector selects parts of the content of a section and stores them in the results object, so that they can be modified in the script. As Control Scripts don't have a selector, the results object can't be used there. Control Scripts don't touch the content - meaning, the text flow - of the sections.

Adding a Control Script

To add a Control Script:

1. On the **Scripts** pane at the bottom left, click the black triangle on the **New** button and click **New Control Script**. A new script appears in the list.
2. Double-click the new script to open it. The script editor appears.
3. Change the name of the script so that it reflects what the script does.
4. Write the script; see the "Control Script API" on page 930. If you are not familiar with scripting, also see "Writing your own scripts" on page 624.

Tip

New Control Scripts added to the template contain code to continue the page numbering over all print sections, and two examples: one to select different sections

of a Print context for email and print output, and one to select a web section.

What to use a Control Script for

Control Scripts let you change the way a template is merged, by giving access to the template with all its contexts and sections in a script. A Control Script may, for example, omit, group and clone sections; add a background to a Print section; or add a header to an email. A number of the things that you can do with them is listed in the table below, with a link to a topic that explains how to do them and that shows what the script should look like.

Control Scripts differ from template scripts in two ways:

- Control Scripts run **before** all other scripts. When a template consists of several contexts, and these contexts are combined in the output - for example, when an Email is generated with the Print context as attachment - all scripts run once for each context, but Control Scripts always go first.
- Control Scripts **don't** touch the content - meaning, the text flow - of the sections. They don't have a selector, like the other scripts do. A selector selects parts of the content of a section and stores them in the `results` object, so that they can be modified in the script. As Control Scripts don't have a selector, the `results` object can't be used there.

In a Control Script, `section` usually is the most important object. To get a quick overview and lots of examples, see "section" on page 936. For help on specific tasks, see the table below.

Task	See topic	Field/function of <code>section</code> object
Change the page numbering of Print sections	"Control Script: Page numbering" on the facing page	<code>restartPageNumbering</code>
Set the background image of a Print section	"Control Script: Setting a Print section's background" on page 653	<code>background.source</code> , <code>background.url</code> , <code>background.position</code>
Split Email attachments and rename them	"Parts: splitting and renaming email attachments" on	<code>part</code>

Task	See topic	Field/function of section object
	page 651	
Dynamically set a password on PDF attachments	"Control Script: Securing PDF attachments" on page 658	<code>password</code> , <code>ownerPassword</code>
Include/exclude sections: <ul style="list-style-type: none"> • Conditionally omit sections • Output one section or another, based on the value of a data field • Select one print section as PDF attachment if the output is to be emailed, and another print section if the output is to be printed. 	Use a Conditional Print Section script to in/exclude a Print section based on a simple condition; see "section" on page 936 In all other cases take a look at the examples in the following topic: "section" on page 936.	<code>enabled</code>
Add sections dynamically	"Dynamically adding sections (cloning)" on page 655.	<code>clone()</code>
Add a header to an email	"section" on page 936, example: "Adding custom ESP handling instructions" on page 978.	<code>headers</code>

Control Script: Page numbering

This topic explains how to write a Control Script that changes the page numbering in Print sections. Note that when you add a Control Script, it already contains a script to make the page numbering continue over all Print sections.

For information about Control Scripts in general, see "Control Scripts" on page 645 and "Control Script API" on page 930. If you don't know how to write scripts, see "Writing your own scripts" on page 624.

How to change page numbering in a control script

A Control Script can make the page numbering continue over all Print sections or let it restart on a section. This is done by setting the `restartPageNumber` field on a section to `true` or `false`. For example: `merge.template.contexts.PRINT.sections['Section 2'].restartPageNumber = true;` (Also see "section" on page 936 and "Control Script API" on page 930.)

Page numbering starts with page 1 for each section. If for a section `restartPageNumber` is set to `false`, that section will start with the page number following the last page of the previous section.

Note that even if a section is not enabled (so it will not be outputted), its `restartPageNumber` flag is still taken into account for composing the page number sequences.

By default, each section has `restartPageNumber = false` when the first control script runs.

Tip

If you are looking to create a table of contents, add a template script that uses the `pageRef()` function. For an example, see "Creating a table of contents" on page 900.

Examples

Restarting the page numbers several times

Assume that a template has four sections (of 1 page each) in the Print context and a Control Script sets the page numbering as follows:

1. Section A (1 page) `restartPageNumber = true`
2. Section B (1 page) `restartPageNumber = true`
3. Section C (1 page) `restartPageNumber = false`
4. Section D (1 page) `restartPageNumber = true`

The code would look like this:

```
if (merge.context.type == ContextType.PRINT) {
    merge.context.sections['Section A'].restartPageNumber = true;
    merge.context.sections['Section B'].restartPageNumber = true;
    merge.context.sections['Section C'].restartPageNumber = false;
    merge.context.sections['Section D'].restartPageNumber = true;
}
```

The page numbering in the output will be:

1. Section A page 1
2. Section B page 1
3. Section C page 2
4. Section D page 1

Disabled section

When a section is disabled, it will not be outputted, but its `restartPageNumber` flag will still be taken into account for composing the page number sequences. So, if the `restartPageNumber` flags are set as follows:

1. Section A (1 page) `restartPageNumber = true`
2. Section B (2 pages) `restartPageNumber = false`
3. Section C (3 pages) `restartPageNumber = true, enabled = false`
4. Section D (4 pages) `restartPageNumber = false`

In code:

```
if (merge.context.type == ContextType.PRINT) {
    merge.context.sections['Section A'].restartPageNumber = true;
    merge.context.sections['Section B'].restartPageNumber = false;
    merge.context.sections['Section C'].restartPageNumber = true;
    merge.context.sections['Section C'].enabled = false;
    merge.context.sections['Section D'].restartPageNumber = false;
}
```

The page numbering in the output will be:

1. Section A page 1
2. Section B page 2

3. Section D page 1 (page numbering is restarted due to section C's restartPageNumber = true)

Parts: splitting and renaming email attachments

In a Control Script, **parts** can be defined to determine which sections should be output to the same file. This way it is possible to split the Print context or the Web context into multiple email attachments. This topic shows how to do that.

For information about Control Scripts in general, see "Control Scripts" on page 645 and "Control Script API" on page 930. If you don't know how to write scripts, see "Writing your own scripts" on page 624.

Defining parts

Defining parts is done by setting the `part` field on a `section`, for example:

```
merge.template.contexts.PRINT.sections['Section 2'].part = "PDF_Attachment2";. (Also see "section" on page 936 and "Control Script API" on page 930.)
```

- If a part name is given, then that delimits the start of a new part (even if the part name is the same as the previous one). Following sections that don't define a part name, will be added to the previous part.
- A part ends at the last enabled* section or at the last section before the start of a new part.
*When a Control Script has set the `enabled` field of a `section` to `false`, it will not be outputted.

If no part name is set on any section, it is assumed that there is only one part, consisting of the default section (for Web output) or of all sections (for Print output). The attachment(s) will be named after the email subject.

Examples

No parts defined

Assume there are three Print sections: sections A, B and C. When generating Email output with the Print context as attachment, all three Print sections will be put together in one file and attached to the email. If the email's subject is 'Take action', the name of the attached file will be 'Take action.PDF'.

Splitting and renaming a Print attachment

Assume there are three Print sections: sections A, B and C. In a Control Script a part name is defined for section C:

```
var section = merge.template.contexts.PRINT.sections['Section C'];
section.part = 'Part2';
```

When generating Email output with the Print context as attachment, the email will have two attachments:

- attachment 1: Section A, Section B
- attachment 2: "Part2", which is Section C. The file name of this attachment is the part name.

Note

For Web sections, a part always consists of only the given section. Web pages cannot be appended to form a single part. It is however possible to attach multiple Web pages to one email; see the following example.

Controlling multiple Email attachments

The following script attaches the following sections to an email:

- Print section 3 + 4 as attachment with continued page numbers
- Print section 6 as separate attachment
- Web sections A and B as separate attachments

```
if (channel == Channel.EMAIL) { // only when generating Email
output
if (merge.context.type == ContextType.PRINT) {
    merge.context.sections['Section 1'].enabled = false;
    merge.context.sections['Section 2'].enabled = false;
    merge.context.sections['Section 3'].enabled = true;
    merge.context.sections['Section 3'].part = "PDFAttach1";
    merge.context.sections['Section 4'].enabled = true;
    merge.context.sections['Section 4'].restartPageNumber = false;
    merge.context.sections['Section 5'].enabled = false;
    merge.context.sections['Section 6'].enabled = true;
    merge.context.sections['Section 6'].part = "PDFAttach2";
```



```
} else if (merge.context.type == ContextType.WEB) {
    merge.context.sections['default Section'].enabled = false; //
disable whatever is the default section
    merge.context.sections['Section A'].enabled = true;
    merge.context.sections['Section A'].part = "WebPartA";
    merge.context.sections['Section B'].enabled = true;
    merge.context.sections['Section B'].part = "WebPartB";
}
}
```

Note

For another example, see this how-to: [Output sections conditionally](#).

Control Script: Setting a Print section's background

In the Print context, a PDF file can be used as a Print section's background. To learn how to do this without a Control Script, see "Using a PDF file as background image" on page 339. With a Control Script, a Print section's background can be set dynamically. You could for example specify a particular PDF file as a section's background depending on the value of a field in the current record. This topic shows how.

For information about Control Scripts in general, see "Control Scripts" on page 645 and "Control Script API" on page 930. If you don't know how to write scripts, see "Writing your own scripts" on page 624.

Setting a background in script

The Control Script should first enable a background on the section, in case an initial background wasn't set via the user interface. This is done by setting the source type for the background of the section to either DataMapper PDF or Resource PDF (see "BackgroundResource" on page 948). For example:

```
merge.template.contexts.PRINT.sections['Policy'].background.source
= BackgroundResource.RESOURCE_PDF;
```

A DataMapper PDF is, as you would expect, a PDF generated by the DataMapper. A Resource PDF is a PDF from another source.

For a DataMapper PDF, nothing else has to be done to set the background. For a Resource PDF, the Control Script should specify a path, for example:

```
var resourceUrl = 'images/policy-' + record.fields.policy + '.pdf';
merge.template.contexts.PRINT.sections['Policy'].background.url =
resourceUrl;
```

Positioning the background

After that, the background can be positioned, setting the section's `background.position`:

```
activeSection.background.position = MediaPosition.FIT_TO_MEDIA;
```

For all possible positions, see "MediaPosition" on page 951.

Setting a page range in script

When a PDF that serves as a dynamic section background has multiple pages, you can specify a range of pages to be used, in a control script.

Put the number of the first page in the range in the section's `background.start` field and the last page in `background.end`.

This requires you to set the `background.allPages` option to `false`, first. This option is `true` by default and takes precedence, so when it is `true`, the entire PDF will be used, even if a page range has been set.

The following script sets the page range from 2 to 5:

```
merge.template.contexts.PRINT.sections
['Policy'].background.allPages = false;
merge.template.contexts.PRINT.sections['Policy'].background.start =
2;
merge.template.contexts.PRINT.sections['Policy'].background.end =
5;
```

Tip

You could use the `resource()` function to check the number of pages or for example the page height and width before setting it as a background (see "resource()" on page 928).

Example

This script sets a background on a Print section using absolute positioning.

```
var activeSection = merge.template.contexts.PRINT.sections['Section 1'];
activeSection.background.source = BackgroundResource.RESOURCE_PDF;
activeSection.background.url = "images/somepage.pdf";
activeSection.background.position = MediaPosition.ABSOLUTE;
activeSection.background.left = "10mm";
activeSection.background.top = "10mm";
```

You could replace the last three lines of the previous script by the following line to scale the Print section background to Media size:

```
activeSection.background.position = MediaPosition.FIT_TO_MEDIA;
```

Dynamically adding sections (cloning)

This topic explains how to clone a section in a Control Script, Print sections can be cloned, so that a document can have a dynamic number of sections, based on data. This is particularly useful when the record set defines one or more PDFs (e.g. insurance policies) per recipient. Via a Control Script, for each PDF a section can be cloned and each clone can be given one of the PDFs as background (see "Control Script: Setting a Print section's background" on page 653). For each page in the PDF, a page will be added to the section.

For information about Control Scripts in general, see "Control Scripts" on page 645 and "Control Script API" on page 930. If you don't know how to write scripts, see "Writing your own scripts" on page 624.

Cloning a section

To clone a section, first use the `clone()` function and then add the clone to the Print context before or after a specific section, using `addAfter()` or `addBefore()`:

```
var printSections = merge.template.contexts.PRINT.sections;
var clone = printSections["Section 1"].clone();
printSections["Section 1"].addAfter(clone);
```

Cloned sections have the same properties as normal sections, but they cannot call `section` functions.

Note

Due to resource constraints, the number of unique clones that can be created throughout a job is limited to around 20. A clone is considered unique if it has a different name. This is a rough estimate; if the template is simple, up to 60 clones may be created.

The limit only applies to the amount of unique clones. There is no limit to the amount of `clone()` function calls.

Renaming a clone

By default, clones receive the name of their source section with a "Clone {sequence}" suffix, for example:

Source: "Section 1"

Clone Name: "Section 1 Clone 1"

Use the `name` property to assign the cloned section another name, for example:

```
clone.name = "my_section_clone";
```

The section name must be unique within the scope of a single record.

Note

It is recommended not to use a random value or a time stamp in the name of a section. Although section names must be unique within the scope of a record, *across records* it is advisable to keep using the same name for the same clone, to avoid hitting the limit of the number of unique clones that can be created throughout a job (see the previous note).

Targeting elements in a cloned section

As each clone receives a unique section name, one could use CSS style sheets (see "Styling and formatting" on page 551) and personalization scripts (see "Variable Data" on page 604 and "Writing your own scripts" on page 624) to further personalize the cloned sections.

The following CSS style rules target the `<h1>` element in a number of clones and assigns the respective text a different color:

```
[section="my_section_clone_0"] h1 { color: red; }
[section="my_section_clone_1"] h1 { color: green; }
[section="my_section_clone_2"] h1 { color: blue; }
```

The same selectors could be used in personalization scripts:

Selector: `[section="my_section_clone_0"] h1`

Script: `results.css('color','red');`

In a template script, cloned sections can be found using `merge.section`:

```
if (merge.section == "my_section_clone_0") {
    results.html("Clone!");
} else {
    results.html("Original.");
}
```

Note that in a Control Script, `merge.section` is only defined when the output channel is WEB; see "merge" on page 935.

Examples

Cloning a section based on the number of records in a detail table

This script creates as many clones of a section as there are records in a detail table. It assigns the new sections a unique name.

```
var printSections = merge.template.contexts.PRINT.sections;
var numClones = record.tables['detail'].length;
for( var i = 0; i < numClones; i++){
    var clone = printSections["Section 1"].clone();
    clone.name = "my_section_clone_" + i;
    printSections["Section 1"].addAfter(clone);
}
```

Cloning a section based on data and assigning a background PDF

This script clones a section based on data fields. It disables the source section first and then calls the `addPolicy` function. `addPolicy` clones the section, renames it and sets a PDF from the resources as its background. It explicitly enables the clone and then adds it to the Print context.

```
var printSections = merge.template.contexts.PRINT.sections;
merge.template.contexts.PRINT.sections["Policy"].enabled = false;
```

```

if(record.fields.policy_a == 1) {
    addPolicy('a');
}
if(record.fields.policy_b == 1) {
    addPolicy('b');
}
function addPolicy(policy){
    var resourceUrl = 'images/policy-' + policy + '.pdf';
    var clone = printSections["Policy"].clone();
    clone.name = "policy_" + policy;
    clone.background.url = resourceUrl;
    clone.enabled = true;
    printSections["Policy"].addAfter(clone);
}

```

Control Script: Securing PDF attachments

The Print context can be attached to an email in the form of a PDF file and secured with a password. This can be done without a Control Script, see "Email attachments" on page 379 and "Email PDF password" on page 378.

With a Control Script, you can do the same, and more: the attachment can be split into multiple attachments (see Parts). Each attachment may have a different (or no) set of passwords, so you could mix secured and unsecured attachments in a single email. This topic shows how.

For information about Control Scripts in general, see "Control Scripts" on page 645 and "Control Script API" on page 930. If you don't know how to write scripts, see "Writing your own scripts" on page 624.

Setting passwords in script

To set a password on a Print section in a Control Script, the script should first retrieve the Print section/s using `merge.template.contexts.PRINT.sections` or `merge.context.sections` (also see the example below).

Next, the script can split the attachments, if needed (see "Parts: splitting and renaming email attachments" on page 651), and it can set a password on each `section`. For example:

- `merge.template.contexts.PRINT.sections['Section 2'].password = 'secret';`
- `merge.template.contexts.PRINT.sections['Section 2'].ownerPassword = 'secret';`

When producing a **single** attachment, the password(s) should be set on the first Print section. When producing **multiple** attachments, it should be set on the first section of each part.

Password types

PDF allows for two types of passwords to be set on a secured PDF file: a user password and owner password. The user password allows a limited access to the file (e.g. printing or copying text from the PDF is not allowed). The owner password allows normal access to the file. The Email PDF password script sets both the user and owner password to the same value, so that when the recipient provides the password, he can manipulate the file without limitations.

In a Control Script:

- `password` is used to set the user password and owner password for a PDF attachment to the same value.
- `ownerPassword` is used to set the owner password for a PDF attachment. Setting only the owner password creates a secured PDF that can be freely viewed, but cannot be manipulated unless the owner password is provided. Note that the recipient needs Adobe Acrobat to do this, because the Acrobat Reader does not allow users to enter the owner password.

Removing a password

Passwords set in the Control Script override the password set through the Email PDF password script (see "Email PDF password" on page 378). This allows you to change or remove the password from a specific part. Removal is done by setting the `password` field to `null` or an empty string ("").

Example

This scripts splits the Print output into two PDF attachments and sets a password for the second attachment.

```
var printSections;
if (channel == Channel.EMAIL) { // only when generating Email
output
if (merge.context.type == ContextType.PRINT) {
printSections = merge.template.contexts.PRINT.sections;
    printSections['Section 1'].part = 'PDFAttach1';
    printSections['Section 2'].part = 'PDFAttach2'
    printSections['Section 2'].password = 'secret';
```

```
}  
    }  
}
```

The script flow: when scripts run

When Connect generates the actual output – letters, web pages or emails -, it opens a record set and merges it with the template. It takes each record, one by one, and runs all scripts for it, in a specific order, as explained below.

First all Control Scripts are executed, in the order in which they appear in the Scripts pane. Control scripts don't touch the content of the sections themselves, but they change the way a template is outputted, for example by selecting or omitting sections from the output (see "Control Scripts" on page 645).

Then the template scripts are executed, once for each section, in the order in which they appear in the Scripts pane.

Template scripts can change the contents of the current section in a template.

This type of script must have a **selector**: text, an HTML element and/or a CSS selector (see "Writing your own scripts" on page 624).

Running a template script starts with looking in the current section for pieces of content that match the script's selector.

Important to note is that **if nothing matches the selector, the script is not executed**.

In a **Print context**, the template scripts in the Scripts pane run once for each section and then for each Master Page (see "Master Pages" on page 350). Next, each processed Master Page is put behind every page to which it should be applied.

Scripts are NOT executed again for every page. Post-pagination scripts currently don't exist in Connect.

Selectors in Connect

Selectors are patterns used to select one or more HTML elements. They were originally developed to be able to define the layout of web pages without touching their content, through Cascading Style Sheets (CSS). In Connect, since each section in a Connect template is in fact an HTML file (see "Editing HTML" on page 466), the very same selectors can be used in style sheets (see "Styling templates with CSS files" on page 553) and template scripts (see "Personalizing Content" on page 592 and "Writing your own scripts" on page 624). Selectors can increase the speed with which a template and data are merged; see "Use an ID as selector" on page 636.

Standard CSS selectors

Selectors are made up of one or more of the following components:

- An **HTML element**. Type the HTML tag without the angle brackets (e.g. `p`) to select all elements of that type (`p` selects all paragraphs).
- A **class**. Type the class name, preceded by a dot, e.g.: `.green`, to select HTML elements with that class.
- An **ID**. Type the ID, preceded by `#`, e.g.: `#intro`, to select an HTML element with that ID.
- An **attribute** of an HTML element. Type the attribute and, optionally, its value, between square brackets, e.g.: `[target]`, to select HTML elements with a matching attribute.
- A **pseudo-class**. For example, `tr:nth-child(even)` selects all even table rows.

These components can be combined in different ways. For example, `p div` selects all paragraphs *inside* `<div>` elements, while `p, div` selects all paragraphs *and* all `<div>` elements.

A complete list of selectors and ways to combine them, and a tool that demonstrates their use can be found at W3Schools: http://www.w3schools.com/cssref/css_selectors.asp.

A **video** about CSS and Script Selectors, can be found here: [Connect with Evie 6 - CSS and Script Selectors](#).

Connect classes and attributes

Connect itself sometimes adds a specific class or attribute to elements in a template. Capture OnTheGo widgets, for example, have a `role` attribute that allows the COTG library to dictate their behaviour. Connect classes and attributes can be used in selectors, as will be explained and demonstrated below.

Connect-specific classes usually are invisible in the Designer. By opening the currently selected section in your default web browser (click the Preview HTML toolbar button) and using the browser's code or source inspector you can see most of the dynamically added classes.

Warning

Avoid using classes with the `__ol` prefix in your selectors. These dynamically added class names may change in future releases of the software.

Section selector

The Designer writes the name of each section to the `section` attribute of the `<html>` element. This attribute can be used in selectors.

Example

The following rule applies formatting to `<h1>` elements in sections of which the name starts with 'Letter':

```
[section^='Letter'] h1 {  
    color: brown;  
}
```

Note

To make scripts run exclusively on certain sections, it is advised to put them in folders and set the execution scope of the scripts in a folder via the folder properties; see "Execution scope" on page 630.

Sheet position selectors

In Print output, pages have a sheet position that depends on the options set in the Sheet Configuration dialog (e.g. the Duplex and Allow Content On options). Connect gives each page - or rather, the "MediaBox" div element on that page - a class depending on their sheet position:

- `.frontside`
- `.backside` (does not apply to simplex documents)
- `.contentpage`
- `.nocontentpage`

The MediaBox contains the Master Page objects and section backgrounds. This means that these classes can only be used to format a Master Page and section background. They do not let you change the formatting of elements residing in the main text flow (e.g. a `<h1>` element on page 3).

Conditionally formatting Master Page objects

The following CSS rule sets the color of <h1> elements on a Master Page when that Master Page is present on the front of a sheet.

```
.frontside h1 {  
    color: green;  
}
```

The next style rule is a bit more specific: it colors <h1> elements on a Master Page when that Master Page is applied to the front of a sheet in Section 1:

```
[section='Section 1'] .frontside h1 {  
    color: green;  
}
```

The following rule hides <h1> elements on the back of a sheet on which no content (from the main text) is allowed.

```
.backside.nocontentpage .h1 {  
    display: none;  
}
```

Print section background selector

When you inspect a Print section in a browser, you will see that it has a <div id="pages"> element as the first child of the <body> element. Inside this <div> there are one or more MediaBoxes: elements with the class `page_mediabox`. Each MediaBox contains the Media, section background and Master Page that apply to one page (see "Media" on page 353, "Master Pages" on page 350 and "Using a PDF file as background image" on page 339).

In the MediaBox, a Print section background is an element with the `ol_pdf_datamapper_input` class. Its `src` attribute references the PDF file that contains the image and its `page` attribute is used to select a specific page in that PDF (as a PDF can contain more than one page). For example:

```

```

You can use the `ol_pdf_datamapper_input` class as a selector to target the section background in a style rule or script.

Placing the section background in front of the Master Page

The stacking order of elements inside each MediaBox, from bottom to top, is:

1. Media
2. Section background
3. Master Page elements

Using the `.page_mediabox` selector, you could change this stacking order and place the section background on top of the elements on the Master Page. Set the z-index property to a value larger than 0 (zero) and add `!important` to make this style rule override the inline style declaration that normally puts the section background behind the Master Page elements:

```
.page_mediabox img.ol_pdf_datamapper_input {  
    z-index: 10 !important;  
}
```

Scaling the section background

The rule below downscales the section background image and keeps it in the centre of the page:

```
.page_mediabox img.ol_pdf_datamapper_input {  
    transform: translate(-50%, -50%) scale(1.5, 1.5) !important;  
}
```

View selectors

In the Designer, sections can be viewed on different tabs: Source, Design, Preview and - if it is a Web section - Live. In each view mode (except Source) a specific CSS class is added to the `<html>` element. The view-specific classes are:

- `.DESIGN`
- `.PREVIEW`
- `.OUTPUT`

`.OUTPUT` is used when viewing the current section on the Live tab or in an external browser, and when generating output.

View selectors allow you to apply formatting to elements in a certain view, for example to highlight or show elements. The Designer itself does this, for example, to highlight all boxes in the Design view, when the Show Edges icon is clicked.

Adding an outline

The following style rule wraps every element that has the class `address-block` with a purple dashed outline in Design mode. The outline is not visible in other views or when outputting the document.

```
.DESIGN .address-block {
    outline: 1px dashed purple;
}
```

Adding a background pattern

The Postcard template wizard (in the Basic Print templates group) uses the `.DESIGN` class to mark areas that are reserved for postal use and should not contain text or images. These areas were added to the Master Page as absolute positioned boxes that have been given the class `clearzone`. The following style rule assigns a background pattern to elements with that class in the Design view:

```
.DESIGN .clearzone {
    background:url
(data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAAAQAAAAECAYAAACp8Z5+
AAAFU1EQVQImWNgQAL/70z7TyqHgYEBANRfDcEzmlBaAAAAAE1FTkSuQmCC)
repeat;
}
```

Note

The pattern image was created on www.patternify.com and is added as a data URI (see [Data URIs](#)).

Showing hidden Foundation elements

In Capture OnTheGo templates based on the Foundation framework the `.DESIGN` selector can be used to show elements that would otherwise be hidden in the Design view.

For example, to expand accordion elements and show validation errors in Design view, you could add the following style rules to your template:

```
.DESIGN .accordion .accordion-navigation > .content {
    display: block;
}
.DESIGN small.error {
```

```
    display: block;
    margin-top: -20px;
}
```

Designer User Interface

The main ingredients in the Designer's user interface are the following:

- "Menus" on page 744
- "Toolbars" on page 771
- "Resources pane" on page 761
- "Outline pane" on page 761
- "Attributes pane" on page 756
- "Styles pane" on page 768
- "Workspace" on page 768
- "Data Model pane" on page 758
- "Scripts pane" on page 766
- "Preflight Results and Messages" on page 759

Dialogs

Dialogs can allow you to perform a command or make settings. They can also ask you a question or provide you with information or progress feedback.

Here is a list of all dialogs:

Bar Chart Properties dialog

The Bar Chart dialog appears when a Bar Chart object is right-clicked and the Bar Chart... option is clicked. It determines how the Bar Chart is displayed when generating output and in Preview mode.

General Tab

- **General Group:**
 - **Display grid above graph:** Check to display the grid on top of the bars so that it is always visible.
 - **Rotate:** Check to rotate the graph 90 degrees so that the bars are horizontal starting from the left.
 - **Stack Series:** [TBD]
- **Text Group:** Determines how text is displayed in labels and legends.
 - **Font:** Type in the font-face to use to display text. The font must be installed on the system and defaults to Verdana if the font is not found. Equivalent to the `font-family` property.
 - **Size:** Type in the size of the font. For example, 12pt or 20px. Defaults to 11px. Equivalent to the `font-size` property.
 - **Color:** Type in the color in which to display text. The color value must be a valid [HTML Hex Color](#). Equivalent to the `color` property.
- **Fill Group:**
 - **Color:** Enter a color for the bars. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#). This color replaces all initial colors.
 - **Opacity:** Enter the percentage for the opacity of the bars. Does work on the initial colors, if no fill color is entered in this dialog.
- **Line Group:**
 - **Show Line:** Adds a line around each bar (or fills the bar if the bar has no fill color).
 - **Color:** Enter a color for the line. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).
 - **Opacity:** Enter the percentage for the opacity of the line.

Value Axis Tab

- **Title group:**
 - **Label:** Enter a label for the Y axis (X axis if the graph is rotated).
 - **Bold:** Check if you want the label to be in bold style.

- **Color:** Enter a custom color for the label (Default is Black). The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).
- **Font Size:** Enter a font size for the label, in pt.
- **Grid group:**
 - **Show Grid:** Displays a grid behind the bars.
 - **Color:** Enter a color for the grid that displays in the graph. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).
 - **Opacity:** Enter the opacity percentage of the grid. Default is 15%.
 - **Thickness:** Enter a thickness for the grid, in pixels. Default is 1px.
 - **Tick Length:** The distance between each vertical line in the grid.
- **Axis group:**
 - **Show Axis:** Check to show the value axis (the line between the chart and the values).
 - **Color:** Enter a color for the value axis. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).
 - **Opacity:** Enter the opacity in percentage for the axis.
 - **Thickness:** Enter the thickness, in pixels, for the axis.

Category Axis Tab

- **Title group:**
 - **Label:** Enter a label for the X axis (Y axis if the graph is rotated).
 - **Bold:** Check if you want the label to be in bold style.
 - **Color:** Enter a custom color for the label (Default is Black). The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).
 - **Font Size:** Enter a font size for the label, in pt.
- **Grid group:**
 - **Color:** Enter a color for the grid that is displays in the graph. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).
 - **Opacity:** Enter the opacity percentage of the grid. Default is 15%.
 - **Thickness:** Enter a thickness for the grid, in pixels. Default is 1px.

- **Position:** Choose Middle to centre the grid over the graph, or choose Start to make the first vertical grid line match the value axis.
- **Tick Length:** The distance between each vertical line in the grid.
- **Axis group:**
 - **Show Axis:** Check to show the value axis (the line between the chart and the values).
 - **Color:** Enter a color for the value axis. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).
 - **Opacity:** Enter the opacity in percentage for the axis.
 - **Thickness:** Enter the thickness, in pixels, for the axis.

Chart Tab

- **3D group:**
 - **Apply 3D Effect:** Check to apply a 3D effect to the bars.
 - **Depth:** Enter a thickness of the 3D effect, in pixels.
 - **Angle:** Enter angle for the 3D effect, in degrees.

Legend tab

- **Show Legend:** Check to show the legends in the chart object.
- **Legend Group:** Defines how the legends are shown.
 - **Equal label widths:** Check so that all labels are of equal width in the Legends box. The Legend's width will accommodate the largest value.
 - **Position:** Use the drop-down to select where the labels are shown: Right, Left, Top or Bottom.
 - **Align:** Use the drop-down to select how to align the text in the labels: Left, Middle or Right.
 - **Horizontal Space:** When multiple columns appear, enter a numerical value (in pixels) to define horizontal spacing between the columns.
 - **Vertical Space:** Enter a numerical value (in pixels) to define vertical spacing between legends.

- **Max Columns:** Enter a numerical value to define the maximum number of columns allowed in the Legends box.
- **Values Group:** Defines if and how values are shown in the Legends box.
 - **Show Values:** Check to show values besides the Legend's label.
 - **Text:** Enter the text used to display the values. Variables can be used to display specific data,
 can be used to create a new line:
 - `[[percents]]` : Contains the percentage of the chart the value represents.
 - `[[value]]` : Contains the numerical value of the field.
 - Any Text: Adding text (such as a dollar sign or column, etc) will make it appear in each label.
- **Markers Group:** Defines how the Legends Markers. Markers are icons with a color matching the Legend with its corresponding bar.
 - **Type:** Use the drop-down to select in which shape the Markers are displayed. Select None to hide the Markers completely.
 - **Size:** Enter the size (in pixels) for the Markers to be displayed.
 - **Label Gap:** Enter the distance (in pixels) between the Markers and the Legends text.
 - **Border Width:** Use the drop-down to define the thickness of the border added to the Markers. Default is 0pt.
 - **Border Color:** Enter a valid HTML Hex Color for the border's color.
 - **Border Opacity:** Enter a numerical value between 0 and 100 to define the opacity (in percentage) of the border.

Box Formatting dialog

The Text Formatting dialog is accessible by clicking inside a box in the template and then selecting **Format > Box** in the menu.

All settings in this dialog are in fact CSS properties. Cascading Style Sheets (CSS) were originally designed for use with web pages: HTML files. Since Designer templates are HTML files, they are styled with CSS. To learn how to use CSS in the Designer, see "Styling and formatting" on page 551 and "Styling templates with CSS files" on page 553. For information about specific properties and their options, see [W3Schools CSS Reference](#).

Note

When no unit is added to a geometry value, such as the width, height, top or margin, the default unit will be added to the value; see "Print Preferences" on page 710.

For information about the different types of Boxes, see "Boxes" on page 513.

Box tab

- **General group:**

- **Width:** Set the width of the box in measure or percentage. When no unit is entered, the default unit will be added to the value (see "Print Preferences" on page 710). Equivalent to the CSS `width` property.
- **Height:** Set the height of the box in measure or percentage. When no unit is entered, the default unit will be added to the value (see "Print Preferences" on page 710). Equivalent to the CSS `height` property.
- **Angle:** Set the rotation angle of the box in clockwise degrees. Equivalent to the CSS `transform: rotate` property.
- **Corner radius:** Set the radius of rounded border corners in measure or percentage. Equivalent to the CSS `border-radius` property.
- **Display:** Use the drop-down or type in the value for how to display the box. Equivalent to the CSS `display` property.
- **Overflow:** Use the drop-down or type in the value for how to handle overflow (text that does not fit in the current size of the box). Equivalent to the `overflow` property.

- **Text wrap group:**

- **Float:** Use the drop-down or type in the value for how to float the box, if the box is not in an absolute position. Equivalent to the CSS `float` property.
- **Clear:** Use the drop-down or type in the value for clearing pre-existing alignments. Equivalent to the CSS `clear` property.

- **Positioning:**

- **Position:** Use the drop-down or type in the value for the type of positioning for the box. Equivalent to the CSS `position` property.
- **Top:** Set the vertical offset between this box and its parent's top position. Equivalent to the CSS `top` property.

- **Left:** Set the horizontal offset between this box and its parent's left position. Equivalent to the CSS `left` property.
- **Bottom:** Set the vertical offset between this box and its parent's bottom position. Equivalent to the CSS `bottom` property.
- **Right:** Set the horizontal offset between this box and its parent's left position. Equivalent to the CSS `right` property.
- **Z-index:** Set the z-index of the box. The z-index defines in which order elements appear. Equivalent to the CSS `z-index` property.

Background tab

For information about backgrounds see "Background color and/or image" on page 579.

- **General group:**
 - **Color:** Specify the color of the box background. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#). Equivalent to the CSS `background-color` property.
- **Background image group:**
 - **Source:** click the **Select Image** button to select an image via the "Select Image dialog" on page 725. Equivalent to the CSS `background` property.
 - **Size:** select `auto`, `cover` or `contain` (for an explanation see http://www.w3schools.com/cssref/css3_pr_background-size.asp), or type the width and height of the image in a measure (e.g. `80px 60px`) or as a percentage of the box size (e.g. `50% 50%`). Equivalent to the CSS `background-size` property.
 - **Position:** select the position for the background-image. Equivalent to the CSS `background-position` property.

Spacing tab

For information about spacing see "Spacing" on page 591.

- **Padding group:** Defines padding (spacing inside the element) in measure or percentage:
 - **All sides:** Check to set all padding to use the Top value. Equivalent to the CSS `padding` property.

- **Top, Left, Bottom, Right:** Set padding for each side. Equivalent to the CSS `padding-left`, `padding-top`, `padding-right` and `padding-bottom` properties.
- **Margin group:** Defines margins (spacing outside the element) in measure or percentage:
 - **All sides:** Check to set all margins to use the Top value. Equivalent to the CSS `margin` property.
 - **Top, Left, Bottom, Right:** Set the margin for each side. Equivalent to the CSS `margin-left`, `margin-top`, `margin-right` and `margin-bottom` properties.

Border tab

For information about borders see "Border" on page 580.

- **Same for all sides:** Defines the border properties for all sides using the Top properties. Equivalent to the CSS `border` property.
- **Top, Left, Bottom, Right:** Each group defines the following properties:
 - **Width:** Specify the thickness of the border. Equivalent to the CSS `border-width` property.
 - **Style:** Specify the style of the border such as `solid`, `dashed` or `dotted`. Equivalent to the CSS `border-style` property.
 - **Color:** Specify the color of the border. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#). Equivalent to the CSS `border-color` property.

Content tab

- **Copy Fit:** Check this option to automatically scale text inside the Box to the available space; see "Copy Fit" on page 566 .
 - **Minimum font size:** Specify the minimum font size using a valid font measurement unit (pt, px, in, cm or mm). Do not put a space between the number and the unit. The smallest possible size is 1pt. The default is 4pt. When left **blank**, the font size in Design view becomes the minimum font size. Text can only be made bigger than its initial size.
 - **Maximum font size:** Specify the maximum font size using a valid font measurement unit (pt, px, in, cm or mm). Do not put a space between the number and the unit. The biggest possible size is 1048pt. The default is 48pt.

When left **blank**, the font size in Design view becomes the maximum font size. Text can only be made smaller than its initial size.


- **Fit to width only**: When this option is checked, no line breaks will be added to the text.
- **Child** (optional): When specified, the Copy Fit feature will only be applied to the given child element (an element inside the Box or Div). Specify the element by giving its ID, for example: **#product**, or class, for example: **.product** - note the dot.

Color Picker

The Color Picker dialog appears when creating a color in the formatting dialogs of certain elements, for example border colors in boxes and paragraphs. For information about how to define and apply colors and how to keep them consistent in different output channels, see "Colors" on page 583.

The dialog consists of two main parts. On the left is the color wheel that can be used to select a color **hue** by clicking anywhere on that wheel. To the right of the color wheel there is a vertical bar used to select the color **saturation**. At the top-right, two colors are shown: the **New** box displays the currently selected color, while the **Original** shows the color currently attributed to the element.

The rest of the dialog has various options for choosing colors:

- **Color Mode**: Use the drop-down to select whether the color is set as **RGB**, **CMYK** or **HEX**. The color mode determines how the color is saved in the formatting properties, and how they are printed or output; see "Colors" on page 583.
- **RGB group**: Enter the Red, Green and Blue color values from 0 to 255.
- **HEX**: Enter a valid [HTML Hex Color](#).
- The **eye dropper** lets you select a color from anywhere on your desktop. To open it, click the eye dropper button  next to the HEX color field.
- **CMYK group** (suitable for Print output only): Enter the Cyan, Magenta, Yellow and Black color values from 0 to 100 percent.

Note

Whenever one value within this dialog is changed, all the other values are adjusted to their equivalent.

Colors Properties

The Colors Properties defines and sets named colors used in the template; see "Colors" on page 583. Named colors can be used throughout the templates, in all contexts. They are visible in color selector dialogs and useable with their names in style sheets; see "Styling and formatting" on page 551.

- **Color Type Selector:** Click and use the drop-down to display which color types to show in the list: All, RGB, CMYK or Spot colors.
- **Color List:** Displays the colors, filtered using the Color Type Selector.
- **New:** Create a new color using the **Edit Color** dialog (see below and see "Colors" on page 583).
- **Edit:** Edit the currently selected color using the **Edit Color** dialog (see below and see "Colors" on page 583).
- **Delete:** Delete the currently selected color.
- **Duplicate:** Duplicate the currently selected color using the name [color]CopyX.

Edit color

You can edit the following color properties.

- **Name:** Enter the name of the color. This name should not contain spaces or special characters.
- **Create name based on values:** Check so that the name is automatically based on the color slider values below.
- **Type:** Use the drop-down to specify which type of color this should be: either a Tint or a Color.
- **Option group:** contains the options for the chosen type. Options change depending on the selected type.

- **Color:**
 - **Model:** This can be either CMYK or RGB.
 - **Spot Color:** Check to set the color as Spot Color. When Spot Colors are used, the Name must match that of the spot color used in the printer.
 - **Cyan/Magenta/Yellow/Black (CMYK):** Each slider sets a percentage for the color. Set the values using the sliders, or type in the percentage directly in the input boxes.
 - **Red/Green/Blue (RGB):** Each slider sets the values of 0-255 for the color. Set the value using the sliders or type in the value directly in the input boxes.
 - **Color Preview:** Box displaying the preview of the color (converted to RGB when relevant).
- **Tint:**
 - **Source:** Select an existing *Color* in the template. The *tint* or *opacity* will be applied to this color.
 - **Tint/Opacity:** The slider sets the percentage of tint or opacity. Set the value using the slider, or type the percentage directly in the input box.
 - **Use Opacity:** Check to set the Tint slider to use Opacity instead.

Color Settings

Color Management can keep colors consistent across different outputs by using Color Profiles. When producing output to a new device, color adjustments are made to present the color as accurately as possible on this new device.

- **Enable Color Management:** Check to disable color management and ignore embedded color profiles when importing images (with the exception of imported PDF files as it might contain a multiple tagged sub images).
- **Working Space Group:** Defines the color profiles for the current template.
 - **RGB:** Use the drop-down to select a color profile for RGB colors. The list displays ICC profiles located in "%USERPROFILE%\Connect\color-profiles\rgb".
 - **CMYK:** Use the drop-down to select a color profile for CMYK colors. The list displays ICC profiles located in "%USERPROFILE%\Connect\color-profiles\cmyk".
 - **Gray:** Use the drop-down to select a color profile for Grayscale. The list displays ICC profiles located in "%USERPROFILE%\Connect\color-profiles\gray".

- **Untagged Images Group:** Defines color profiles for any image that does not specifically have color profiles or color settings enabled.
 - **RGB:** Use the drop-down to select a color profile for RGB colors. The list displays ICC profiles located in "%USERPROFILE%\Connect\color-profiles\rgb".
 - **CMYK:** Use the drop-down to select a color profile for CMYK colors. The list displays ICC profiles located in "%USERPROFILE%\Connect\color-profiles\cmymk".
 - **Gray:** Use the drop-down to select a color profile for Grayscale. The list displays ICC profiles located in "%USERPROFILE%\Connect\color-profiles\gray".
- **Options Group:**
 - **Rendering intent:** Use the drop-down to specify how colors are converted that are out of range of a profile. For example, you may use tricks like reducing the saturation of the entire print so that a color that is out of range still appears a bit more vibrant than ones that are in range. Rendering intents use different methods to trick the eye into believing that the print can reproduce irreproducible colors.

Edit Label Properties

The Edit Label Properties defines how a Pie Chart Label displays its title and data. It contains two options:

- **Label:** Enter a title for Labels and Legends when they are shown (see "Pie Chart Properties dialog" on page 692).
- **Value:** Use the drop-down to select which Value to use as data within the Pie Chart as well as for Label and Legend values.

Find/Replace Dialog

The Find/Replace dialog can replace text within the current template. The scope of the replacement depends on the currently selected tab in the Workspace. If the Source tab is selected, the replace will affect the HTML source code. If the Design tab is selected, the replace will affect the text on the page. If the Preview tab is selected, the Replace feature is inactive.

Note

When replacing text in the Design tab, formatting in the replaced text will be removed. If formatting is necessary in the new text, select the Source tab before opening the Find/Replace dialog and include the required HTML tags in the replacement text.

Here are the options available in this dialog

- **Find:** The source string to find.
- **Replace with:** The string to replace the source with.
- **Direction**
 - **Forward:** Look forward from the current position of the pointer in the template or source.
 - **Backward:** Look backward from the current position of the pointer in the template or source.
- **Scope**
 - **All:** Searches in the complete text of the template or source.
 - **Selected lines:** Searches in the currently selected text or source.
- **Options**
 - **Case sensitive:** Use a case-sensitive search, which differentiates **TEXT** from **text** or **Text**.
 - **Wrap search:** Loop back from the end of the template or selection to its beginning, when the Search is at the end of the template or the selection.
 - **Whole word:** Searches for the source string as a whole word.
 - **Incremental:** With this option selected, each letter you type in the Find field causes the editor focus to move to the first complete occurrence of the text you are typing.
 - **Regular expressions:** Enables regular expressions for a search in the **Source** view of the workspace. After checking this option, you can type Ctrl + Space in either text box to view a list of regular expressions.

Tip

The Find/Replace dialog can fill in regular expressions in the Find field by itself. Open the dialog, check the option Regular expressions and close the dialog again. Select the text you want to search for and reopen the dialog: the Find field will now contain the regular expression for the text to find.

- **Find:** Click to find the next instance of the source string.

- **Replace/Find:** Click to replace the current instance with the replacement text and go to the next instance of the source string.
- **Replace:** Click to replace the current instance with the replacement text.
- **Replace All:** Click to replace all instances of the source string with the replacement text.
- **Close:** Close the dialog.

Font Manager

The Fonts Manager contains the fonts that were added to the template manually. It essentially lists the fonts located in the **Fonts** folder of the Resources pane (see "Fonts" on page 587).

Fonts with the same file name with a different extension are considered variations of the same font. For example, if there are three files, named gotham-book-webfont.eot, gotham-book-webfont.ttf, gotham-book-webfont.woff, only "gotham-book-webfont" appears in the Name column of this dialog.

The following buttons appear to the right of the list of fonts:

- **New:** Click to open the Edit Font dialog to add a new font.
- **Edit:** Click to open the Edit Font dialog to edit the currently selected font.
- **Remove:** Click to delete the currently selected font entry.
- **Duplicate:** Click to create a copy of the currently selected font entry.

Edit Font

The Edit Font dialog appears when clicking New or Edit from the Fonts Dialog.

- **Name:** Enter the name that should be used to refer to the font. This is equivalent to the `font-family` property of the `@font-face` CSS rule (see http://www.w3schools.com/cssref/css3_pr_font-face_rule.asp).
- **Font Weight:** Use the drop-down to select the default font weight (the thickness, see http://www.w3schools.com/cssref/pr_font_weight.asp):
 - **None:** Does not define the property.
 - **Normal:** Defines font-weight as normal
 - **Bold:** Defines the font-weight as bold (equivalent to a numerical value of 700).
 - **Numerical values:** Defines the line thickness; 400 is normal, 700 is bold.

- **Font Style:** Use the drop-down to select the font style (see http://www.w3schools.com/cssref/pr_font_font-style.asp):
 - **None:** Does not define the property.
 - **Normal:** Defines font-style as normal
 - **Italic:** Makes the font italic.
 - **Oblique:** Makes the font oblique (this is generally the same as italic but does not require a special italic version of the font).
- **Name:** Check the fonts in the list to include them in the font definition.

Image Formatting dialog

The Image Formatting dialog is accessible by selecting an image in the template and then selecting **Format > Image** in the menu.

All settings in this dialog are in fact CSS properties. Cascading Style Sheets (CSS) were originally designed for use with web pages: HTML files. Since Designer templates are HTML files, they are styled with CSS. To learn how to use CSS in the Designer, see "Styling and formatting" on page 551 and "Styling templates with CSS files" on page 553. For information about specific properties and their options, see [W3Schools CSS Reference](#).

For more information about the use of images, see "Images" on page 537 and "Styling an image" on page 576.

Image Tab

- **General group:**
 - **Width:** Set the width of the image in measure or percentage. Equivalent to the CSS `width` property.
 - **Height:** Set the height of the image in measure or percentage. Equivalent to the CSS `height` property.
 - **Angle:** Set the rotation angle of the image in clockwise degrees. Equivalent to the CSS `transform: rotate` property.
 - **Corner radius:** Set the radius of rounded border corners in measure or percentage. Equivalent to the CSS `border-radius` property.

- **Display:** Use the drop-down or type in the value for how to display the image. Equivalent to the CSS `display` property.
- **Overflow:** Use the drop-down or type in the value for how to handle overflow (the part of the image that does not fit in the current size of the box). Equivalent to the CSS `overflow` property.
- **Source:** Enter the web address or local file address of the image. Equivalent to the HTML `src` attribute.
- **Alternate text:** Enter an alternate text for the image. This is displayed in browsers and email clients when the image is loading or if the image cannot be displayed. It is also used for accessibility. Equivalent to the HTML `alt` attribute.
- **Text wrap group:**
 - **Float:** Use the drop-down or type in the value for how to float the image, if the image is not in an absolute position. Equivalent to the CSS `float` property.
 - **Clear:** Use the drop-down or type the value to clear pre-existing alignments. Equivalent to the CSS `clear` property.
- **Positioning:**
 - **Position:** Use the drop-down or type in the value for the type of positioning for the image. Equivalent to the CSS `position` property.
 - **Top:** Set the vertical offset between this image and its parent's top position. Equivalent to the CSS `top` property.
 - **Left:** Set the horizontal offset between this image and its parent's left position. Equivalent to the CSS `left` property.
 - **Bottom:** Set the vertical offset between this image and its parent's bottom position. Equivalent to the CSS `bottom` property.
 - **Right:** Set the horizontal offset between this image and its parent's left position. Equivalent to the CSS `right` property.
 - **Z-index:** Set the z-index of the image. The z-index defines in which order elements appear. Equivalent to the CSS `z-index` property.

Spacing Tab

- **Padding group:** Defines padding (spacing inside the element) in measure or percentage:

- **All sides:** Check to set all padding to use the Top value. Equivalent to the CSS `border` property.
- **Top, Left, Bottom, Right:** Set padding for each side. Equivalent to the CSS `border-left`, `border-top`, `border-right` and `border-bottom` properties.
- **Margin group:** Defines margins (spacing outside the element) in measure or percentage:
 - **All sides:** Check to set all margins to use the Top value. Equivalent to the CSS `margin` property.
 - **Top, Left, Bottom, Right:** Set the margin for each side. Equivalent to the CSS `margin-left`, `margin-top`, `margin-right` and `margin-bottom` properties.

Border Tab

- **Same for all sides:** Defines the border properties for all sides using the Top properties. Equivalent to the CSS `border` property.
- **Top, Left, Bottom, Right:** Each group defines the following properties:
 - **Width:** Specify the thickness of the border. Equivalent to the CSS `border-width` property.
 - **Style:** Specify the style of the border such as `solid`, `dashed` or `dotted`. Equivalent to the CSS `border-style` property.
 - **Color:** Specify the color of the border. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#). Equivalent to the CSS `border-color` property.

Includes dialog

The Includes dialog defines which style sheets and JavaScript files should be applied to a section when generating output (see: "Styling templates with CSS files" on page 553 and "Using JavaScript" on page 403).

To open this dialog and make settings for one section, right-click the section on the **Resources** pane and select **Includes**.

To do this for all Web sections at the same time, right-click the Web context on the Resources pane, or select **Context > Includes** on the main menu. (This menu option is only available when a Web section is being edited in the Workspace.)

Email clients do not read CSS files and some even remove a <style> tag when it is present in the email's header. Nevertheless, CSS files can be used with the Email context in the Designer. When generating output from the Email context, the Designer converts all CSS rules that apply to the content of the email to inline style tags, as if local formatting was applied.

1. From the **File types** dropdown, select **Stylesheets, JavaScripts** or **all**.
2. The list at the left displays the style sheets and/or JavaScript files that are present in the template's resources. The list at the right shows the style sheets and or JavaScript files that will be included in the output of the current section (or in all Web sections, if you are making settings for the Web context). Use the **Include** and **Exclude** buttons to move files from one list to the other.
3. Files are included in the order shown. To change this order, click one of the included files and use the **Up** or **Down** button.

Note

The styles in each following style sheet add up to the styles found in previously read style sheets. When style sheets have a conflicting rule for the same element, class or ID, the **last** style sheet 'wins' and overrides the rule found in the previous style sheet.

Line Chart Properties dialog

The Line Chart dialog appears when a Line Chart object is right-clicked and the Line Chart... option is clicked. It determines how the chart is displayed when generating output and in Preview mode.

General Tab

- **General Group:**
 - **Display grid above graph:** Check to display the grid on top of the lines so that it is always visible.
 - **Rotate:** Check to rotate the graph 90 degrees so that the lines are vertical starting from the top.
 - **Stack Series:** Stack the lines so that lines representing the same value do not overlap.

- **Text Group:** Determines how text is displayed in labels and legends.
 - **Font:** Type in the font-face to use to display text. The font must be installed on the system and defaults to Verdana if the font is not found. Equivalent to the `font-family` property.
 - **Size:** Type in the size of the font. For example, 12pt or 20px. Defaults to 11px. Equivalent to the `font-size` property.
 - **Color:** Type in the color in which to display text. The color value must be a valid [HTML Hex Color](#). Equivalent to the `color` property.
- **Fill Group:**
 - **Color:** Enter a color for the lines. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#). This color replaces all initial colors.
 - **Opacity:** Enter the percentage for the opacity of the bars. Does work on the initial colors, if no fill color is entered in this dialog.
- **Line Group:**
 - **Show Line:** Adds a line around each line part (or fills the line part if it has no fill color).
 - **Color:** Enter a color for the line. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).
 - **Opacity:** Enter the percentage for the opacity of the line.

Value Axis Tab

- **Title group:**
 - **Label:** Enter a label for the Y axis (X axis if the graph is rotated).
 - **Bold:** Check if you want the label to be in bold style.
 - **Color:** Enter a custom color for the label (Default is Black). The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).
 - **Font Size:** Enter a font size for the label, in pt.
- **Grid group:**
 - **Color:** Enter a color for the grid that is displays in the graph. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).
 - **Opacity:** Enter the opacity percentage of the grid. Default is 15%.

- **Thickness:** Enter a thickness for the grid, in pixels. Default is 1px.
- **Tick Length:** The distance between each vertical line in the grid.
- **Axis group:**
 - **Show Axis:** Check to show the value axis (the line between the chart and the values).
 - **Color:** Enter a color for the value axis. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).
 - **Opacity:** Enter the opacity in percentage for the axis.
 - **Thickness:** Enter the thickness, in pixels, for the axis.

Category Axis Tab

- **Title group:**
 - **Label:** Enter a label for the X axis (Y axis if the graph is rotated).
 - **Bold:** Check if you want the label to be in bold style.
 - **Color:** Enter a custom color for the label (Default is Black). The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).
 - **Font Size:** Enter a font size for the label, in pt.
- **Grid group:**
 - **Color:** Enter a color for the grid that is displays in the graph. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).
 - **Opacity:** Enter the opacity percentage of the grid. Default is 15%.
 - **Thickness:** Enter a thickness for the grid, in pixels. Default is 1px.
 - **Position:** Choose Middle to centre the grid over the graph, or choose Start to make the first vertical grid lign match the value axis.
 - **Tick Length:** The distance between each vertical line in the grid.
- **Axis group:**
 - **Show Axis:** Check to show the value axis (the line between the chart and the values).
 - **Color:** Enter a color for the value axis. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#).

- **Opacity:** Enter the opacity in percentage for the axis.
- **Thickness:** Enter the thickness, in pixels, for the axis.

Legend tab

- **Show Legend:** Check to show the legends in the chart object.
- **Legend Group:** Defines how the legends are shown.
 - **Equal label widths:** Check so that all labels are of equal width in the Legends box. The Legend's width will accommodate the largest value.
 - **Position:** Use the drop-down to select where the labels are shown: Right, Left, Top or Bottom.
 - **Align:** Use the drop-down to select how to align the text in the labels: Left, Middle or Right.
 - **Horizontal Space:** When multiple columns appear, enter a numerical value (in pixels) to define horizontal spacing between the columns.
 - **Vertical Space:** Enter a numerical value (in pixels) to define vertical spacing between legends.
 - **Max Columns:** Enter a numerical value to define the maximum number of columns allowed in the Legends box.
- **Values Group:** Defines if and how values are shown in the Legends box.
 - **Show Values:** Check to show values besides the Legend's label.
 - **Text:** Enter the text used to display the values. Variables can be used to display specific data,
 can be used to create a new line:
 - `[[percents]]` : Contains the percentage of the chart the value represents.
 - `[[value]]` : Contains the numerical value of the field.
 - Any Text: Adding text (such as a dollar sign or column, etc) will make it appear in each label.
- **Markers Group:** Defines how the Legends Markers. Markers are icons with a color matching the Legend with its corresponding line.
 - **Type:** Use the drop-down to select in which shape the Markers are displayed. "none" hides the Markers completely.
 - **Size:** Enter the size (in pixels) for the Markers to be displayed.

- **Label Gap:** Enter the distance (in pixels) between the Markers and the Legends text.
- **Border Width:** Use the drop-down to define the thickness of the border added to the Markers. Default is 0pt.
- **Border Color:** Enter a valid HTML Hex Color for the border's color.
- **Border Opacity:** Enter a numerical value between 0 and 100 to define the opacity (in percentage) of the border.

Locale Settings

The Locale dialog box sets the locale used inside the template. The Locale can affect time, currency output, and other formatting that depends on location and language (see "Locale" on page 590). The default Locale for new templates can be set via the Preferences ("Language Setting Preferences" on page 710).

- **Use:** Use the drop-down to select how the Locale is set for the current template.
 - **System Locale:** Select this to use the operating system's locale settings. This is set in the Region settings of the control panel.
 - **Explicit Locale:** Select this option to specify a static locale which will remain static for this template, whichever server the template is used on.
 - **Data Field:** Select this to use a data field from the record. The locale will be record-specific in this case.
- **Locale:** Use the drop-down to select a specific locale. Only enabled when **Explicit Locale** is selected above.
- **Data Field:** Use the drop-down to select a field within the current data model that contains the locale. This field must be a string and contain the exact locale to be used, such as "en" or "fr-CA". It cannot be an alias such as "english" or "french". The locale supports both ISO-639-1 alone ("en", "fr", etc) or ISO-639-1 followed by a 2-letter country code ("de-DE", "zh-CN", "fr-CA", "fr-FR", etc).

Master Page Properties

Master Pages can only be used in a Print context; see "Master Pages" on page 350.

The following properties are available for Master Page resources:

- **Name:** The name of the master page, displayed in all drop-downs where the Master Page is shown as well as in the "Resources pane" on page 761.
- **Margins** group:
 - **Header:** The space at the top of the Master Page where no content will print, when this Master Page is used in a Section.
 - **Footer:** The space at the bottom of the Master Page where no content will print, when this Master Page is used in a Section.

Media Properties

Media can only be used in a Print context. For an explanation of what Media are and how to use them, see "Media" on page 353.

Media are not printed, unless you want them to; see "Printing virtual stationery" on page 359.

To open the Media properties dialog, right-click one of the Media in the Media folder on the **Resources** pane and select **Properties**.

Properties Tab

- **Name:** The name of the media, displayed in all drop-downs where the Media is shown as well as in the [Resources Pane](#).
- **Size** group: This group is read-only and only used to display the size selected in the linked Print section's properties (see "Print section properties" on page 720).
 - **Page Size:** The named page size.
 - **Width:** The width of the page.
 - **Height:** The height of the page.
 - **Orientation:** Whether the page is portrait or landscape.

Virtual Stationery Tab

- **Front/Back group:** Defines the preprinted media used for the front and back of the Virtual Stationery.
 - **PDF:** Click the Select Image button to open the "Select Image dialog" on page 725 and select which PDF (and optionally, which page of the PDF) to display as a

background for the page.

- **Position:** Use the drop-down to select how the PDF is displayed on the page:
 - **Fit to Media:** Select to stretch the PDF to fit the media size.
 - **Centered:** Select to center the PDF on the page, vertically and horizontally.
 - **Absolute:** Select to place the PDF at a specific location on the page. Use the Top and Left options below to specify the positioning of the PDF.
 - **Top:** The distance between the top side of the page and the top side of the PDF.
 - **Left:** The distance between the left side of the page and the left side of the PDF.
- **Front side:** Select the image that is shown as a background for all "front" sides in the template.
- **Back side:** Select the image that is shown as a background for all "back" sides in the template.

Characteristics tab

The characteristics define the type of paper on which the Print context is meant to be printed.

- **Media Type:** The type of paper, such as *Continuous, Envelope, Labels, Stationery, etc.*
- **Weight:** The weight of the media in grammage (g/m²).
- **Front Coating:** The pre-process coating applied to the front surface of the media, such as *Glossy, High Gloss, Matte, Satin, etc.*
- **Back Coating:** The pre-process coating applied to the front surface of the media.
- **Texture:** The texture of the media, such as *Antique, Calenared, Linen, Stipple or Vellum.*
- **Grade:** The grade of the media, such as *Gloss-coated paper, Uncoated white paper, etc.*
- **Hole Name:** Pre-defined hole pattern that specifies the pre-punched holes in the media, such as *R2-generic, R2m-MIB, R4i-US, etc.*

Paragraph Formatting dialog

The Paragraph formatting controls how the selected paragraph is formatted. It is accessed by placing the cursor within a paragraph and then selecting **Format > Paragraph** on the menu.

All settings in this dialog are in fact CSS properties. Cascading Style Sheets (CSS) were originally designed for use with web pages: HTML files. Since Designer templates are HTML files, they are styled with CSS. To learn how to use CSS in the Designer, see "Styling and formatting" on page 551 and "Styling templates with CSS files" on page 553. For information about specific properties and their options, see [W3Schools CSS Reference](#).

For information about text and how to style it see "Text and special characters" on page 546 and "Styling text and paragraphs" on page 562.

Formats tab

- **General group:**

- **Line-height:** Specify the height of each line in the element's text, in measure or percentage. Note that this is not spacing between lines, but rather the complete height of the line itself including the text. Equivalent to the `line-height` property.
- **Align:** Select how text should be aligned, such as `left`, `center`, `right` or `justify`. Equivalent to the `align` property.
- **First Indent:** Specify the indentation of the first line of each paragraph in the element. Equivalent to the `text-indent` property.
- **Display:** Select how to display the element. This can also be used to hide an element completely using the `none` option. See [CSS Display](#). Equivalent to the `display` property.
- **Direction:** Select in which direction text should be displayed (`ltr`, `rtl`, `auto`). Useful for certain languages such as arabic, hebrew, etc. Equivalent to the `dir` HTML attribute.

- **Breaks group:**

- **Before:** Specifies how to handle page breaks before the element. Equivalent to the `page-break-before` property.
- **Inside:** Specifies whether to accept page breaks within the paragraph. Equivalent to the `page-break-inside` property.
- **After:** Specifies how to handle page breaks after the element. Equivalent to the `page-break-after` property.
- **Widows:** Specifies how to handle widows within the paragraph (lines appearing alone on the next page if the paragraph does not fit on the current one). Equivalent to the `widows` property. Widows and orphans are ignored if the `page-break-inside` property is set to `avoid`.

- **Orphans:** Specifies how to handle orphans within the paragraph (lines appearing alone at the end of a page if the paragraph does not fit on the current one). Equivalent to the `orphans` property.

Note

For more information on page breaks, widows and orphans, see the [W3 Paged Media reference](#).

Spacing tab

- **Padding group:** Defines padding (spacing inside the element) in measure or percentage:
 - **All sides:** Check to set all padding to use the Top value. Equivalent to the CSS `padding` property.
 - **Top, Left, Bottom, Right:** Set padding for each side. Equivalent to the CSS `padding-left`, `padding-top`, `padding-right` and `padding-bottom` properties.
- **Margin group:** Defines margins (spacing outside the element) in measure or percentage:
 - **All sides:** Check to set all margins to use the Top value. Equivalent to the `margin` property.
 - **Top, Left, Bottom, Right:** Set the margin for each side. Equivalent to the `margin-left`, `margin-top`, `margin-right` and `margin-bottom` properties.

Border tab

- **Same for all sides:** Defines the border properties for all sides using the Top properties. Equivalent to the `border` property.
- **Top, Left, Bottom, Right:** Each group defines the following properties:
 - **Width:** Specify the thickness of the border. Equivalent to the `border-width` property.
 - **Style:** Specify the style of the border such as `solid`, `dashed` or `dotted`. Equivalent to the `border-style` property.
 - **Color:** Specify the color of the border. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#). Equivalent to the `border-color` property.

PDF Attachments dialog

The PDF Attachments dialog defines options for the Email context that are used when generating email output with PDF attachments (see "Generating Email output" on page 973).

To open this dialog, right-click the **Email** context on the **Resources** pane and select **PDF attachments**.

Alternatively, select **Context > PDF Attachments** on the main menu. This option is only available when an Email section is being edited in the Workspace.

- **Print Context Image Compression:** Defines the properties of the PDF when attaching the Print context to email output.
 - **Lossless:** Enables maximum quality in the PDF. Note that this will produce a larger PDF.
 - **Quality:** Disabled when Lossless is checked. Determines the quality (aka compression) of the attached PDF.
 - **Tile Size:** Use the drop-down to select the size of the tiles used in the image. When low Quality values are used to optimize images smaller than 1024 x 1024 pixels, using the largest tile size will produce better results.

Pie Chart Properties dialog

The Pie Chart dialog appears when a Pie Chart object is right-clicked and the **Pie Chart...** option is clicked. It determines how the Pie Chart is displayed when generating output and in Preview mode (see "Business graphics" on page 516).

General tab

- **Text Group:** Determines how text is displayed in labels and legends.
 - **Font:** Type in the font-face to use to display text. The font must be installed on the system and defaults to Verdana if the font is not found. Equivalent to the `font-family` property.
 - **Size:** Type in the size of the font. For example, 12pt or 20px. Defaults to 11px. Equivalent to the `font-size` property.
 - **Color:** Type in the color in which to display text. The color value must be a valid [HTML Hex Color](#). Equivalent to the `color` property.

- **Slice Colors Group:** Determines which colors are used to display the Pie Chart.
 - **Apply:** select which set of colors to use for the chart: **standardColors**, **baseColor** (a set of colors based on the color defined in the Base Color option) or **colors** (a set of colors defined in the Color Array option).
 - **Base Color:** Enter a valid HTML Hex Color. When a Base Color is set, it will be the color of the first slice and the colors of all other slices are based on this color.
 - **Brightness Step:** Enter the amount of brightness to change on each new slice. Positive values increase brightness (max: 100), minimum values decrease brightness (minimum: -100). Default is 10.
 - **Color Array:** Enter a comma separated list of hex colors to specify the colors of the slices. If there are more slices than colors in this list, the chart picks random colors. Example: #FF3300, #FFFF00, #33CC33, #FFCC00.
 - **Gradient Ratio:** Enter a start and end point gradient to be applied to each slice (for example: -0.5, 0.5).
- **Slice Outline Group:** Determines whether an outline should be added to each slice of the chart.
 - **Width:** Use the drop-down to select the width of the outline for each pie slice. Values are 0pt, 0.5pt, 1pt, 1.5pt, 2pt or 3pt.
 - **Color:** Enter a valid HTML Hex Color for the outline to appear.
 - **Opacity:** Enter the opacity of the outline. 100 is fully opaque, 0 is transparent.

Pie tab

- **Pie Group:** Defines how the pie chart is displayed in the template.
 - **Automatically calculate radius:** Check to automatically calculate the radius of the Pie Chart, determined by the size of the object it is contained in. The radius, by default, is 50% of the shortest length of the containing <div> object.
 - **Radius:** Enter the radius of the Pie Chart in percentage of the shortest length of the containing <div> object.
 - **Hole Radius:** Enter the radius of the center of the Pie Chart to remove, between 0% and 100%. The hole radius removes the center of the chart, creating a doughnut hole pie chart.

- **Start Angle:** Enter the starting angle of the first slice of the chart, between 0 and 360. This essentially rotates the Pie Chart. Note that if a 3D effect is added to the chart, the only accepted values are 90 or 270 degrees.
- **3D Group:** Defines 3D effects of the Pie Chart.
 - **Apply 3D effect:** Check to enable the Pie Chart to be displayed in a 3D fashion.
 - **Depth:** Enter a numerical value for the thickness of the Pie Chart. Must be in steps of 10 (0, 10, 20, etc).
 - **Angle:** Enter the angle at which the Pie Chart is rotated to create the 3D effect. Default is 20 degrees of rotation.

Labels tab

- **Hide Labels:** Check to disable the label's display.
- **Labels Group:** Defines how the label text is shown.
 - **Custom label text:** Check to enable custom text for the labels. The default display is `[[title]]: [[percents]]`.
 - **Text:** Enter the text to use to display labels. Variables can be used to display specific data, `
` can be used to create a new line:
 - `[[title]]` : Contains either the contents the Label column if Static Labels are used, or the Field Name if Dynamic Labels are used.
 - `[[percents]]` : Contains the percentage of the Pie Chart the value represents.
 - `[[value]]` : Contains the numerical value of the field.
 - Any Text: Adding text (such as a dollar sign or column, etc) will make it appear in each label.
 - **Radius:** Enter a numerical value representing the percentage of the Pie Chart's radius to add as a space between the pie and the labels. The value can be negative, in which case labels are shown within the Pie Chart. If a positive value is used, a line (called a "tick") from each slice of the pie to its label is added.
- **Tick Group:** Defines how ticks (line between the Pie Chart and its labels) is shown.
 - **Color:** Enter a valid HTML Hex Color for the color of the tick.
 - **Opacity:** Enter a percentage of opacity for the tick to be displayed. Default 20 (20% opacity).
- **Grouping Group:** Defines how smaller percentage are grouped together into an individual "Other" category.

- **Apply slice grouping:** Check to enable grouping.
- **Less than %:** Enter a percentage below which values are placed within the "Other" category.
- **Slice Title:** Enter a name for the label of the "Other" category. Defaults to "Other".
- **Color:** Enter a valid HTML Hex Color for the slice. Defaults to the colors set in the General tab.

Legend tab

- **Show Legend:** Check to show the legend in the Pie Chart object.
- **Legend Group:** Defines how the legend is shown.
 - **Equal label widths:** Check so that all labels are of equal width in the Legend box. The Legend's width will accommodate the largest value.
 - **Position:** Use the drop-down to select where the labels are shown: Right, Left, Top or Bottom.
 - **Align:** Use the drop-down to select how to align the text in the labels: Left, Middle or Right.
 - **Horizontal Space:** When multiple columns appear, enter a numerical value (in pixels) to define horizontal spacing between the columns.
 - **Vertical Space:** Enter a numerical value (in pixels) to define vertical spacing between legends.
 - **Max Columns:** Enter a numerical value to define the maximum number of columns allowed in the Legends box.
- **Values Group:** Defines if and how values are shown in the Legends box.
 - **Show Values:** Check to show values besides the Legend's label.
 - **Text:** Enter the text used to display the values. Variables can be used to display specific data,
 can be used to create a new line:
 - `[[percents]]` : Contains the percentage of the Pie Chart the value represents.
 - `[[value]]` : Contains the numerical value of the field.
 - Any Text: Adding text (such as a dollar sign or column, etc) will make it appear in each label.
- **Markers Group:** Defines how the Legends Markers. Markers are icons with a color matching the Legend with its corresponding Pie Chart slice.

- **Type:** Use the drop-down to select in which shape the Markers are displayed. "none" hides the Markers completely.
- **Size:** Enter the size (in pixels) for the Markers to be displayed.
- **Label Gap:** Enter the distance (in pixels) between the Markers and the Legends text.
- **Border Width:** Use the drop-down to define the thickness of the border added to the Markers. Default is 0pt.
- **Border Color:** Enter a valid HTML Hex Color for the border's color.
- **Border Opacity:** Enter a numerical value between 0 and 100 to define the opacity (in percentage) of the border.

Preferences

The Preferences dialog is used to modify the general software preferences. Changes made in this dialog affect the software globally, not individual templates and data mapping configurations.

The Preferences dialog is separated into individual tabs, where each tab controls certain aspects of the software.

To open the Preferences dialog, select **Window > Preferences**.

General preferences

The General Preferences are as follows:

- **Always run in background:** This option correlates with the "Always run in background" option selectable in the "Document Boundaries Refresh" dialog and "Print via Print Server" dialog. When either of these dialogs is used and the option is checked, it will also be checked here. To prevent the refresh boundaries and print via print server dialogs to automatically run as background, uncheck this option.

COTG Servers

By allowing one or more Capture OnTheGo servers to be set up, this option anticipates the release of a Capture OnTheGo On Premise Server product.

- **Name:** Enter a unique name.
- **URL:** Enter a valid URL (including the protocol, e.g. http://).
- **Restore Defaults:** Removes all custom servers from the list and resets to the default Capture OnTheGo server.

Clean-up Service preferences

The Clean-up Service defines how the Connect database and the temporary files created during Connect production runs are cleaned up after the production run has finished.

Note

As part of the job production process PlanetPress Connect uses a database for intermediate storage and also creates various temporary "managed" files. These files include data extractions, configuration files and any intermediate files created during the production process. Connect keeps track of all these files through references held within the Connect database.

All the files created and the database references to them are stored for a set amount of time in order to allow Connect to reuse them. However, we do not want to store these indefinitely, because the database would run out of space. The solution is to use the "Clean-up Service" to remove the temporary data and files once they are no longer needed. This clean up service is usually managed by the Server Engine.

Tip

The more items that are present in the database, and the larger they are, the more time and processing power (CPU) that will be required for cleaning them up. Thus a regular Clean-up of the database (as often as possible) is recommended.

This is especially the case if items are not going to be retrieved from the database at a later date.

i.e. If the Connect job is not going to be re-run.

The clean-up can always be set to run outside of business hours (see the **Run**

according to the cron schedule option below), to reduce impact upon Production systems.

The values below define when the specified targets are to be *set* as being ready for deletion, not *when* they are actually deleted. The actual deletion occurs only when PlanetPress Connect is started (if **Run at application start up** is selected), or when the **Run Now** button is pressed, or as per the cron job scheduling.

- **Enable clean-up service:** Check to enable the Clean-up services. When checked, either or both of the *Database clean-up* and *File clean-up* services can be set individually. If the box is not checked, then no Clean-up will occur.
- **Run at application start up:** Click to start the clean-up service when the Designer module is opened, or the Managing Service is started.
- **Run according to the cron schedule:** Enter the interval at which the Clean-up service runs.

To understand how to write a cron job schedule, please refer to the excellent [Quartz Scheduler](#) reference page.

Note

If the **Product managing the service** is set to Designer, then the Designer *must be running* at the time that the cron job is scheduled, for the Clean-up to run.

- **Product managing the service:** Select which of the applications will run the service.

Note

The **Server Engine** is set as the default as it is generally considered the best option.

This is particularly the case when using a scheduled cron job, as the Sever Engine is always running, whilst the Designer might well not be at the scheduled time (in which case the clean-up will fail to run).

- **Database Clean-up Service:**

- **Allow database clean-up service:** Select this checkbox to enable the database Clean-up settings, and enable the actual clean-up.
- **Threads to use for database deletions:** The number of Threads to be used in the clean-up. PlanetPress Connect is a multi-threaded application, and the clean-up is likewise.

Tip

The default number of threads is considered the best compromise for running both clean-up and production jobs simultaneously. If experience suggests that the clean-up is not running efficiently, then upping the number of threads here would be recommended. Conversely, if production appears to be suffering courtesy of the clean-up process, then reduce the number of threads here.

In general, higher end machines (those with multiple cores) will allow a higher numbers of threads, whilst low end machines will perform better with a lower number of threads.

- **Number of entities in each deletion batch:** The number of entities to be deleted at a time. This is done to break the clean-up into smaller chunks. This improves PlanetPress Connect clean-up responsiveness, whilst the clean-up is occurring.
The number selected here applies to all the following settings.
i.e. a selection of 1,000 would delete 1,000 data records within a **Data Set**, 1,000 content items within a **Content Set**, and so on.
- **Minimum time to retain Data Sets:** The minimum time a Data Set (and all the records it contains) is retained within the database before being set for deletion.
- **Minimum time to retain Content Sets:** The minimum time a Content Set (and all the content items it contains) is retained within the database before being set for deletion.

- **Minimum time to retain Job Sets:** The minimum time a Job Set (and all the jobs information it contains) is retained within the database before being set for deletion.
- **Minimum time to retain Managed Files:** The minimum time file references (to files such as data mapping configurations and templates) are retained within the database before being set for deletion.
- **Minimum time to retain other entities:** The minimum time any orphaned data (such as Finishing tables, Media tables, DataModels and Properties tables) are retained within the database before being set for deletion.
- **Database Partition Settings:**
 - **Use Database Partitioning:** Select to use Database Partitioning.
 - **Empty partition count:** The number of empty partitions that are created each clean up run. This defaults to 24.
 - **Partition Size:** Enter the length of time before partitions are switched. This can be entered in minutes, hours, days, weeks or months.
- **File Clean-up Service:**
 - **Allow file clean-up service:** Check to automatically detect orphan files and set them for deletion. Orphan files could be resources and internal files used by Connect, but which are not needed by any running job.
 - **Minimum time to retain orphaned files:** The minimum time during which orphaned files are kept in the database before being set for deletion.

Database Connection preferences

Dialog used to change the PlanetPress Connect back-end Database.

This dialog supports the swapping of the back-end database between various vendor databases. Note, however, that the alternate vendor database(s) *must already be installed and available* in order to swap to them.

Note

This is not a migration tool. It is a simple connection tool, that enables shifting to a different back-end database. Any existing data will **not** be transferred/ migrated between the databases, and any existing Filestores will be cleansed by the [Clean-up](#) Service after

the swap.

Note

When shifting to a different back-end database, the changes won't be applied until PlanetPress Connect is restarted. Including the Connect services. A full machine restart is recommended, as this provides the cleanest restart of all the services.

- **Basic Connection Settings** selections:

- **Database vendor:** Select the database type from the drop down list.

Note

Moving from one vendor database to another will reset all screen selections to defaults, regardless of what may have been previously selected.

- **Database URL:** This is a read-only summation of the current database connection settings.

Tip

If the **Test Connection** button shows that the database cannot be successfully connected to using the selected settings, then the contents of this field could be used to try to connect to the database outside of PlanetPress Connect. This should help in determining and refining the acceptable connection options.

- **Hostname:** Enter the IP Address or alias of the server where database resides.
- **Port:** Enter Port number. The defaults are those which the vendors use by default.

- **Schema:** The individual database schema, within the vendor database.

Note

If a previously non-existent schema were chosen here, then a new schema of that name will be created within the database when the back-end database swap is applied. The tables within that schema, though, will not be created until Connect is restarted.

- **Username:** Enter the database login username.

Tip

It is considered best practice for this user to have root privileges.

- **Password:** Enter the password associated with selected username.
- **Confirm password:** Re-enter the user password.
- **Advanced Connection Settings** selections:
 - **Maximum concurrent threads:** This option sets the maximum database threads. The maximum setting is determined by the specific capabilities of the machine Connect is installed upon (CPU speed and the amount of cores being the major determinants).

Tip

Leaving this value set to the default maximum *should* be the best option in most circumstances.



We recommended this entry be left at the default value.

- **Custom database parameters** table: These are extra parameters which are appended to the database connection URL. The default values are those which have been determined to be useful in connecting to specific vendor databases.

- **Property:** These are free field text fields.

Note

These fields and their associated values get appended to the JDBC connection and therefore must follow all rules regarding acceptable URL addresses for such.

- **Value:** The value applied to the associated Property.
-  **Add:** Used to add extra Property values to the table.
-  **Delete:** Used to remove existing Property values from the table.
- **Test Connection:** Use to test if current connection settings will connect to the specified database.
- **Restore Defaults:** Will restore the settings to PlanetPress Connect HyperSQL standard defaults.
- **Apply:** When a database connection is confirmed as correct this button becomes active, and is used to actually apply the database swap.

Datamapper preferences

Datamapper XML Preferences

- **Display New Line Character as ¶ :** Check to show line returns as ¶ in the Data Viewer, when XML files are shown. If the option is unchecked, you will not see spaces and line returns after element names in the Data Viewer.

Datamapper Default Format Settings

Datamapper stores user preferences for the Date, Number and Currency formats. By default, the user preferences are set to the system preferences. These user preferences become the default format values for any newly created data mapping configuration.

Format settings can also be defined at the data mapping configuration level ("Data mapping configurations" on page 101) and/or per field in the Data Model. Any format settings specified in an existing field are always used, regardless of the user preferences or data source settings.

- **Negative Sign Before** : A negative sign will be displayed before any negative value.
- **Decimal Separator** : Set the decimal separator for a numerical value.
- **Thousand Separator** : Set the thousand separator for a numerical value.
- **Currency Sign** : Set the currency sign for a currency value.
- **Date Format** : Set the date format for a date value.
- **Date Language** : Set the date language for a date value (ex: If English is selected, the term May will be identified as the month of May).
- **Treat empty as 0** : A numerical empty value is treated as a 0 value.


Editing preferences

These preferences define different editing options in the Designer module.

- **Object Resizing for <div> elements**: This defines in which contexts to enable the resizing of <div> elements (including Positioned and Inline boxes). Resizing <div> elements may cause layouts to produce undesirable results especially when using Foundation templates.
 - **Enable for Print Context**: Check to enable <div> resizing in the Print contexts.
 - **Enable for Web Context**: Check to enable <div> resizing in the Web contexts.
 - **Enable for Email Context**: Check to enable <div> resizing in the Email contexts.

Color options

Many of the colors in the user interface of Connect Designer can be adjusted. Click the small colored square next to the field that holds the default color value, to open the Color dialog and pick a color (see "Color Picker" on page 674).

- **Show edges**: The edges around elements in a section in the Workspace. Click the Show Edges icon  to toggle the visibility of these edges.
 - **Box Objects**: This color highlights positioned boxes, inline boxes and Div elements; see "Boxes" on page 513.
 - **Table**: This color highlights tables, and the rows and columns in tables; see "Table" on page 542.
 - **Resizable Table**: This color highlights tables for which the option Allow resizing has been checked when adding the table; see "Table" on page 542.

- **Forms:** This color highlights forms; see "Forms" on page 527.
- **Shared Content:** This color highlights shared content, such as shared snippets; see "Snippets" on page 548.
- **Margin and guides:** These settings only apply to Print sections.
 - **Guides:** This is the color for rulers that can help position content correctly; see "Guides" on page 568.
 - **Margins:** This color delineates the content area on a page; see "Pages" on page 343.
 - **Bleed box:** This color delineates the printable area on a page; see "Page settings: size, margins and bleed" on page 344.
- **Master pages:** These edges are only visible on Master pages; see "Master Pages" on page 350.
 - **Header and Footer Margin:** This color highlights the header and footer margin set for the Master page; see "Adding a header and footer" on page 352.
 - **Objects:** This color highlights all elements on the Master page.
- **Script Result Highlighter:**
 - **Results:** Hovering over a script in the Scripts pane highlights content that will be affected by the script; see "Personalizing Content" on page 592.

Images preferences

- **Transparent PDF image preview:** Check this option so that PDF resources added to the template (including in the Master Page and Media) display using transparency. Note that this can affect display performance (showing transparent PDFs is slower) but will not affect output speed.

Email Preferences

Email (General) Preferences

- **Default From Group:**
 - **Name:** Enter the name that is set by default in the "From name" field in the Send Email and Send Test Email dialogs ("Send (Test) Email" on page 723).
 - **Email Address:** Enter the email that is set by default in the "From Email" field in the Send Email and Send Test Email dialogs ("Send (Test) Email" on page 723).

- **Litmus account Group:**

- **Email Test address:** If you have a Litmus account, enter the test address to use when sending a test email (see "Send (Test) Email" on page 723). For more information on Litmus, please see <http://litmus.com/>.

Email (SMTP) Preferences

SMTP server presets can be selected when sending emails using either the Send Email or Send Test Email dialog. (See "Send (Test) Email" on page 723 and "Email header settings" on page 373). For all presets, the password is not saved and must be re-entered when sending emails.

- The **Add**, **Edit** and **Delete** buttons let you create and manage the presets.
- **SMTP Host Settings:** These settings can be made or edited after clicking the Add or Edit button.
 - **Name:** The name of the preset. This will show up in the Send Email dialog.
 - **Host:** The SMTP server through which the emails are to be sent. Can be a host (mail.domain.com) or an IP address.
 - **Port:** The specified port number will be added to the host name, for example: smtp.mandrillapp.com:465.
 - **Use authentication:** Check if a user name and password are needed to send emails through the host.
 - **Start TLS:** Enabled if authentication is checked. Sends emails through Transport Layer Security (TLS), which is sometimes referred to as SSL.
 - **User:** Enter the user name used to connect to the SMTP server.
- **Restore Defaults:** There are three default presets, each for working with a different Email Service Provider (ESP): Mandrilapp.com, Sendgrid and Mailgun (see "Using an ESP with PlanetPress Connect" on page 976).
- **Apply:** Apply the new settings without closing the Preferences dialog.

Emmet Preferences

Emmet is a framework that enables the lightning-fast creation of HTML code through the use of a simple and effective shortcut language resembling CSS Selectors (see "Emmet" on page 362). The Emmet functionality is available in the HTML and CSS source editors of Connect Designer. Emmet transforms abbreviations for HTML elements and CSS properties to the respective source code.

This is, for example, the abbreviation for a `<div>` element with the class `row`:

```
div.row
```

On pressing the Tab key, this abbreviation is transformed to:

```
<div class="row"></div>
```

To learn more about Emmet itself, please see their website [Emmet.io](https://emmet.io) and the [Emmet.io documentation](#).

Note

Emmet is a plugin. All options listed below are Emmet's default options. They are not specifically adjusted for Connect.

Common Emmet preferences

- **Expand abbreviations by Tab key:** Check to enable the [Expand Abbreviation](#) function.
- **... in files with extension:** Enter a comma-separated list of all file extensions in which expand abbreviation will work.
- **Upgrade web editors:** This Emmet option doesn't affect how Emmet works in Connect Designer.
- **Extensions Path:** Choose a folder where to put json and js files to extend Emmet. This includes custom snippets, preferences and syntax profiles. For more information see [Customization](#).

Emmet Abbreviation Preferences

This Preferences tab lets you add and manage custom abbreviations. All standard abbreviations can be found in Emmet's documentation: [Abbreviations](#).

If there is no need to transform the text while expanding it, create an Emmet snippet instead (see below).

- **New:** Add a new abbreviation.
 - **Name:** The name of the abbreviation is also its trigger.
 - **Context:** The context in which the abbreviation is enabled (HTML, CSS, etc.).
 - **Description:** A short description of the abbreviation .
 - **Pattern:** This defines what an abbreviation expands to. Since Emmet is mostly used for writing HTML/XML tags, abbreviation definition uses XML format to describe elements; see [Abbreviation types](#).
 - **Automatically insert:** This standard option doesn't affect how Emmet works in Connect Designer.
- **Edit:** Edit the currently selected abbreviation.
- **Remove:** Remove the currently selected abbreviation.
- **Import:** Click to open a browse dialog to import an XML file containing exported abbreviations. The imported abbreviations are added to the current list.
- **Export:** Click to open a Save as dialog to export all the abbreviations in an XML file that can be shared and re-imported.
- **Preview box:** Shows what the selected abbreviation is expanded to.
- **Restore Defaults:** clear all custom abbreviations.
- To temporarily disable an abbreviation, uncheck the checkbox next to the name of the abbreviation in the list.

Emmet Output Preferences

The Output Preferences dialog is used to control how the expanded (output) code behaves when expanding abbreviations and snippets. There are 6 different dialogs to control output and, while they all have identical options, they control different output types: CSS, HAML, HTML, XML, XSL and the "Default" one controlling the rest of the types.

These options are equivalent to [Emmet's syntaxProfiles.json feature](#).

Emmet Snippets Preferences

Emmet Snippet are similar to abbreviations in that they are expanded when the Tab key is pressed, but they are just blocks of plain text. Anything in a snippet will be outputted "as is", without any transformation.

- **New:** Click to create a new snippet.
 - **Name:** The name of the abbreviation is also its trigger.
 - **Context:** The context in which the snippet is enabled (HTML, CSS, etc.).
 - **Description:** A short description of the snippet.
 - **Pattern:** The pattern defines what a snippet expands to.
 - **Automatically insert:** This option doesn't affect how Emmet works in Connect Designer.
- **Edit:** Modify the currently selected snippet.
- **Remove:** Remove the currently selected snippet from the list.
- **Import:** Click to open a browse dialog to import an XML file containing exported snippets. The imported snippets are added to the current list.
- **Export:** Click to open a Save as dialog to export all the snippets in an XML file that can be shared and re-imported.
- **Preview box:** Shows what the selected snippet is expanded to.
- To temporarily disable a snippet, uncheck the checkbox next to the name of the snippet in the list.

Emmet Variables Preferences

Variables are placeholders used in Emmet snippets to output predefined data. For example, the `html:5` snippet of HTML syntax has the following definition:

```
<!doctype html>\n<html lang="\${lang}">...</body>\n</html>
```

In the example above, `\${lang}` is used to refer `lang` variable defined in `variables` below. If your primary language is, for example, Russian, you can simply override `lang` variable with `ru` value and keep the original snippets. Also, you can override variable values with inline abbreviation attributes: `html:5[lang=ru]`.

- **Name:** The name of the variable. This should be a single alphanumeric string with no spaces or special characters. For example, the `myVar` name is referred to as `\${myVar}`.
- **Value:** The value of the variable when the snippet is expanded.
- **New:** Click to create a new variable and define its name and value.
- **Edit:** Click to modify the currently selected Variable.
- **Remove:** Click to delete the currently selected Variable.

Language Setting Preferences

- **Display language:** Select a language from the drop-down list to be used as the language of the User Interface (after the software is restarted).
- **Default Locale:** The default locale sets the locale for new templates. By default this is the system's locale. The locale can be changed per template; see "Locale" on page 590.
 - Select System Locale to use the operating system's locale settings.
 - Select Explicit Locale to choose a static locale from the drop-down list.

Print Preferences

Available Printers Preferences

The Available Printers preferences control which printer definitions are available when generating print output or creating Output Presets. Any printer that is unchecked in this dialog will not be visible in the "Model" drop-down of the Print Options dialog; see "Print Options" on page 851 and "Adding print output models to the Print Wizard" on page 960.

Available Printer Preferences:

- **Selected Printers:** Lists the available Printer Definition Files in the system. Note that these are not installed Windows printers or printer queues, but PlanetPress Connect Printer Definition Files.
- **Printer checkbox:** This checkbox selects/deselects all printers in the list. Click to check all, click again to uncheck all.

General Print Preferences

The General Print Preferences are used to set communication settings with the PlanetPress Connect Server module that does the actual generation of print output. The Server module can be located on the same computer (hostname: localhost) or on a different machine. Multiple Designer modules can use a single Server module to generate Print output, as long as the appropriate hostname, username and password are provided. In essence, this can be used to create a single Print Server.

- **Print Server Settings group:**
 - **Protocol:** Use the drop-down to select whether to use the HTTP or the secure HTTPS protocol to connect to the Print Server.

- **Hostname:** Enter the IP, machine name or URL of the Print Server. Default is `localhost`.
- **Port:** Enter the port through which to communicate with the Print Server. Default is `9340`.
- **Username:** Enter the username to authenticate to the Print Server. Default is `ol-admin`. This is set on the server's "Server Security Settings" on page 90.
- **Password:** Enter the password to authenticate to the Print Server. Default is `secret`.
- **Confirm Password:** Re-enter the password above.
- **External sort command timeout (seconds):** Enter the number of seconds to wait for an external sort command before giving up. External sort commands are set up in the [Sorting Options](#) page of the Output module.

Print Measurements Preferences

- **Units:** Use the dropdown to specify the default measurements system used for dimensions of the template and boxes. In addition it defines the coordinates/position of box elements.
The default unit will be added automatically when geometry values are entered without a unit in the Attributes pane or in the Box Properties dialog.
- **Flip insert guide axis:** Check this option to flip the axis on which guides are inserted. Normally, dragging a guide from a horizontal ruler inserts a horizontal guide (see "Guides" on page 568). With this option checked, dragging a guide from a horizontal ruler inserts a vertical guide.

Saving Preferences

The saving preferences are a way control if and how often PlanetPress Connect saves your work in the background, and if how many backup files it creates when you save the template or data mapping configuration. See also: "Saving a template" on page 306.

Auto Save

After a template or data mapping configuration has been saved for the first time, Connect Designer can auto save it with a regular interval.

- **Enable:** activate the Auto Save function.
- **Interval (minutes):** enter a number of minutes, e.g. 3 to auto-save the template or data mapping configuration every 3 minutes.

Auto Backup

Connect Designer can automatically create a backup file when you **manually** save a template or data mapping configuration. The Auto Save function does **not** cause backup files to be created.

- **Enable:** activate the Auto Backup function.
- **Revisions to keep:** Enter the maximum number of backup files. When the maximum is reached, Auto Backup will overwrite the oldest file.
- **Destination:** Select the directory in which the backups should be stored.
 - **Original:** the directory in which the original file is stored.
 - **Other directory:** use the **Browse** button to select another directory.

Backup files have the same name as the original file with two underscores and a progressive number (without leading zeros) at the end: **originalname__1.OL-template, originalname__2.OL-template**, etc.

Scheduling Preferences

The scheduling preferences are a way to control precisely how the PlanetPress Connect services work in the background.

Scheduling options

This preference page defines what is considered a small or large job (anything in between is considered "medium" jobs). For a detailed description of all options, see "Scheduling Preferences" on page 85.

Scheduling - Merge engine

This preferences page defines how different instances and speed units are attributed to different jobs when creating output documents. For a detailed description of all options, see "Merge Engine Scheduling" on page 86.

Scheduling - Weaver engine

This preference page determines the number of engines launched, as well as their speed, when generating Print Output of any type. For a detailed description of all options, see "Weaver Engine Scheduling" on page 88.

Scripting Preferences

The Scripting preferences define different options related to scripting within PlanetPress Connect. See also: "Testing scripts" on page 632.

- **General:**
 - **Script timeout at design time (sec):** In Preview mode or when running the Script Profiler (see the [Profile Scripts](#) dialog), a long running script is stopped after the amount of time set here. The default is 2 seconds, the minimum is 1 second.
- **Designer scripting profiling group:**
 - **Number of iterations:** Enter the number of times to run scripts when running the [Profile Scripts](#) dialog. The default is 1000. Accepted values are 1 to 1000000000. Yes, that's 1 billion - which would take a *long* time to run!

Web Preferences

Web Form Preferences

These preferences define the default behavior of some form elements.

The preferences are as follows:

- **Insert Form Field Defaults:**
 - **Style:** Defines how labels are added to input form elements:
 - **Wrap input with label:** The label is wrapped around the element, such as `<label>First Name <input type="text" name="first_name"></label>`
 - **Attach label to input:** The label is placed before the input, and refers to it: `<label for="first_name">First Name</label> <input type="text" name="first_name">`
 - **Use label as placeholder:** The label is removed and the text is put as a placeholder, such as: `<input type="text" name="first_name" placeholder="First`

Name">

- **No label:** The label value is ignored.
- **Insertion Point:** Defines how new elements are inserted, by default:
 - **At cursor position:** The element is inserted where the cursor is located in the template.
 - **Before element:** The element is inserted before the current element where the cursor is located. For example if the cursor is within a paragraph, insertion occurs before the <p> tag.
 - **After start tag:** The element is inserted within the current element, at the beginning, just after the start tag.
 - **Before end tag:** The element is inserted within the current element, at the end, just before the end tag.
 - **After element:** The element is inserted after the current element where the cursor is located. For example if the cursor is within a paragraph, insertion occurs after the <p> tag.
- **Get Job Data File:** Defines the Workflow URL to be used when the **Get Job Data File on submit** toolbar button is active. This simplifies the process of creating and testing COTG Forms (see "Capture OnTheGo" on page 406).
 - **Workflow URL:** The default URL is: `http://127.0.0.1:8080/_getSampleFormData_`

Profile Scripts dialog

The Script Profiler is accessible through the **Context > Profile Scripts** menu option. It runs the scripts in the template, using the current record, in order to verify the speed at which scripts in the "Scripts pane" on page 766 execute. It helps greatly in troubleshooting performance issues caused by scripting (see also: "Testing scripts" on page 632).

When the dialog opens, the script profiler runs automatically, on 1000 instances of all the scripts by default (this can be changed through the "Scripting Preferences" on the previous page).

Note

The script profiler can take a while, so please be patient.

The results are shown as follows (the first in the line is indicated as **Total** and represents the totals of all the scripts underneath, representing a good overview of the scripts performance in the template):

- **Name:** The name of the script being executed.
- **Count:** As the profiler runs, Count shows the current number of iterations that have been run. This goes up to the total number of set instances and then stops. Hover with your mouse to display a tooltip indicating in which sections the scripts has run (and in which contexts).
- **Elapsed:** Displays the total elapsed time since the start of the session. The table entries are initially sorted based on the values in this column, from high to low. Hovering the mouse over it will display a tooltip that indicates the breakdown of the execution time across different execution stages.
- **Delta:** Displays the estimated difference in performance between the current session and the previous session. Uses average values, so should still work if the previous session was stopped after a different number of iterations. Will be empty if no previous data is available. Hover with your mouse to display a tooltip indicating the breakdown of the execution time across different execution stages.

Script wizards

Wizard types

Script wizards are simplified interfaces for common scripts in templates:

- **Text Script:** This is the default script that is created when a data field is dragged from the Data Model onto the page. See "Variable Data" on page 604.
- **Dynamic Image Script:** Provided that its selector refers to an image, this script dynamically changes the image for each record. See "Dynamic Images" on page 617.
- **Email Scripts.** The Email To Script is automatically added to any new Email context; it defines where the email should be sent for each record. Other Email scripts define other recipients, the subject of the email that is sent, and the PDF password. See "Email header settings" on page 373.
- **Barcode Script:** This script controls the contents of a Barcode. It is automatically added when a barcode is added to a template. See "Barcode" on page 470
- **Business Graphic Script:** This script controls the contents of a Pie Chart, Bar Chart or Line Chart.

The result of the script can be either text appearing on the page, an email address or subject, the barcode data, or a JSON string that is written to the attribute of an HTML element.

Options

Here are the options visible in Script wizards:

- **Name:** The name of the script, making it easier to identify it.
- **Find:** The Selector or Text to apply the result of the script to.
- **Selector:** Uses CSS selectors to find the element to which the script applies
- **Text:** Uses text as a trigger for the script. The script applies to all instances of the text found in the template.
- **Wizard Results:** Displays a list of the data that is sent to replace the content that matches the script's selector:
 - **Prefix:** Static text to use before the set field. For example in Dynamic Image scripts, the default prefix is **images/**.
 - **Field:** A drop-down to select which field contents to use in the script. The field should contain a valid value. For an email script, for example, the field would have to contain an email address. Note that you can't select a field that belongs to a detail table.
 - **Format:** A special formatting modifier applied to the Field; see "Formatting variable data" on page 610.
 - **Suffix:** Static text to use after the set field. For Dynamic Image Scripts, the default suffix is **.jpg** and refers to the file extension.
- **[+]:** Adds a new line to the Wizard Results. Note that by default there is no line return between fields in the list. Adding `
` in the Suffix or Prefix field can establish a line return.
- **[-]:** Removes the currently selected line in the Wizard Results list.
- **Arrow Up:** Moves the currently selected line up one position.
- **Arrow Down:** Moves the currently selected line down one position.
- **Options** (only available in the Text Script wizard and the Dynamic Image wizard): specifies where and how the script inserts its results:
 - As **HTML**. HTML elements in the results are processed and displayed as HTML elements. For instance, `this is bold` will be displayed as **this is**

bold. This is the default setting.

- As **text**. This inserts the results as-is, meaning HTML tags and elements are displayed as text in the output. In this scenario, "
" shows up in the text and does not insert a line break.
- As the value of an **attribute** of an HTML element. The selector of the script should be an HTML element. Which attributes are available depends on the selected HTML element. If the script's selector is an image (element) for example, and the attribute is `src`, the script will modify the image's source. The script's results should be a valid value for the chosen attribute.
- When checked, the option **Convert fields to JSON string** writes the results from the script into an attribute or text as a JSON string. This is useful for Web contexts where a front-end script can read this value easily.
- **OK**: Click to save any changes made to the script, apply the changes in the template, and close the dialog.
- **Cancel**: Click to close the dialog without saving changes.
- **Expand**: Click to convert the script generator to a regular script. Note that this action is not reversible once the regular script has been saved.
- **Apply**: Saves changes made to the script and applies the changes in the template without closing the dialog.

Expanded Script window

When expanded, the Script window replaces all parts of the wizard below the Selector by a box in which the script can be typed. See "Writing your own scripts" on page 624.

For an overview of keyboard shortcuts that can be used in this script editor, see "Keyboard shortcuts" on page 738.

Chart Script dialog

These are the options in the Chart Script dialog:

- **Name**: The name of the script, making it easier to identify it.
- **Find**: The Selector or Text to apply the result of the script to.

- **Selector:** Uses CSS selectors to find the element to which the script applies.
- **Text:** Uses text as a trigger for the script. The script applies to all instances of the text found in the template.
- **Selector and Text:** Uses text as a trigger for the script but only applies to text within the specified Selector.
- **Input Type:** Use the drop-down to select the source of the data to add to the Chart. The selection changes the options below:
 - **Static Labels:** Select to use a static number of data lines below. The chart will always have the same number of items.
 - **Data List:** Lists the data lines that are part of the Chart. Each line represents a segment of the pie as well as a label if they are shown.
 - **Labels:** The text of the label to display next to the Chart or within the legends.
 - **Values:** The value that will be used to create the Chart. This is the name of a field within the Data Model.
 - **Add:** Click to add an entry to the Data List. Opens the [Edit Label Properties](#) dialog.
 - **Delete:** Click to delete the currently selected line in the Data List.
 - **Move Up:** Click to move the currently selected line up one position.
 - **Move Down:** Click to move the currently selected line down one position.
 - **Dynamic Labels:** Select to use data from a detail table to fill the Chart dynamically. At least one detail table must be available in the [Data Model Pane](#) for this option to be functional.
 - **Details:** Use the drop-down to select which detail table provides the data for the Chart.
 - **Labels:** Use the drop-down to select which field within the detail table contains the text for the labels shown in the Chart.
 - **Values:** Use the drop-down to select which field within the detail table contains the numerical values used to build the Chart.

Conditional script dialog

Conditional script generators can show or hide elements on the page depending on certain conditions and values. They can be added by right-clicking any element in a template and

clicking **Make Conditional**. If the current element does not have an ID, one will be automatically generated. See "Showing content conditionally" on page 613.

The options in the Conditional Script wizard are:

- **Name:** The name of the script, making it easier to identify it.
- **Selector:** The Selector or Text to apply the result of the script to.
 - **Selector:** Uses CSS selectors to find the element to which the script applies.
 - **Text:** Uses text as a trigger for the script. The script applies to all instances of the text found in the template.
 - **Selector and Text:** Uses text as a trigger for the script but only applies to text within the specified Selector.

For more information about Selectors see "Using the Text Script Wizard" on page 607.

- **Action:** Use the drop-down to select whether to **Show** or **Hide** the element when the condition below is true.
- **Data Field:** Use the drop-down to select which data field in the record the condition will be based on.
- **Condition:** Select which kind of condition is applied. Possible options are: **Equal to**, **Not equal to**, **Contains**, **Does not contain**, **Begins with**, **Ends with**.
- **Value:** The value used for the conditional check.

For example, you could check whether the value in the data field "Gender" is "Equal To" the value Mr, in order to show a paragraph or an image applying only to male customers.

Section properties dialogs

Email section properties

For information about Email sections, see: "Email templates" on page 370.

Properties tab

The properties for an Email section are minimal and contain the following options:

- **Name:** Enter the name of the Section in the Email Context. This has no effect on output.

Includes tab

This tab lists the style sheets that can be applied to the email section when producing the output. Style sheets are loaded in the order shown, and styles in later style sheets overwrite earlier ones when the same selector is used. (See also: "Styling templates with CSS files" on page 553.)

Print section properties

For information about Print sections, see: "Print sections" on page 335.

Properties

- **Section group:**
 - **Name:** The name of the section.

For an explanation of other settings on the Properties tab, see "Page settings: size, margins and bleed" on page 344.

Includes Tab

The Includes tab defines which style sheets and JavaScript files should be applied to the section when generating output, and in which order; see "Includes dialog" on page 682. For more information about stylesheets, see "Styling templates with CSS files" on page 553. For more information about using JavaScript, see "Using JavaScript" on page 403.

Finishing tab

This tab defines finishing options for this section when it is printed. For an explanation of all Binding and Hole making options, see "Finishing Options" on page 841.

Sheet Configuration tab

The Sheet Configuration tab defines how different Print context sections output on different Media (see "Media" on page 353) and using different Master Pages (see "Master Pages" on page 350). For an explanation of all settings on this tab, see "Sheet Configuration dialog" on page 726.

Background tab

This tab defines the background of the current Print section; see "Using a PDF file as background image" on page 339.

Numbering tab

The Numbering tab defines how page numbers are configured in the current Print section; see "Configuring page numbers" on page 346.

Web Section Properties

The Web section properties define some of the web page properties, especially details appearing in the header.

For information about Web sections, see "Web pages" on page 387.

Properties tab

For information about the settings on the Properties tab, see "Setting the title, meta data and a shortcut icon" on page 392.

Includes Tab

The Includes tab defines which style sheets and JavaScript files should be included in the section when generating output, and in which order; see "Includes dialog" on page 682.

For more information about stylesheets, see "Styling templates with CSS files" on page 553.

For more information about using JavaScript, see "Using JavaScript" on page 403.

Arrange Sections

The Arrange dialog is used to change the order of sections within a context. This can have an effect on how they are outputted; see: "Print sections" on page 335, "Email templates" on page 370 and "Web pages" on page 387.

To access the Arrange dialog, on the Resources pane, right-click on any section or the context containing them, and click **Arrange**.

- **Name:** Displays the name of each section within the context.
- **Move Up:** Click to move the currently selected section up one position.
- **Move Down:** Click to move the currently selected section down one position.

Send COTG Test

The Send COTG Test dialog is used to send a document to the Capture OnTheGo app without the need to go through the Output to Capture OnTheGo task in PlanetPress Workflow (see Workflow Help: [Output to Capture OnTheGo](#)).

For more information about Capture OnTheGo see: "Capture OnTheGo" on page 406. For more information about testing COTG templates see "Testing a Capture OnTheGo Template" on page 436 and this how-to: [Testing a COTG template](#).

Note that to the contrary of the Output to Capture OnTheGo task in Workflow, when using the Send COTG Test dialog, the *contents* of the file will be sent to the Nu-Book server. The Output to Capture OnTheGo task only sends document meta data to the Nu-Book server. The contents of the file are transmitted (by Workflow) when requested by the COTG app.

A test template automatically appears in the app's Repository for 4 days from the moment it is sent. Once downloaded it remains accessible in the app's Library for 2 days.

Note

The dialog is only available on templates containing a Web context. It does not, however, verify whether any Capture OnTheGo form elements have been added to the page.

The dialog contains the following options:

- **General group:**
 - **Server:** Select the Capture OnTheGo server. COTG servers can be added via the preferences; see "COTG Servers" on page 696.
 - **Store ID:** The Nu-Book Store ID. If you don't have one, you can get a trial account for this purpose; please see this page for more details: <http://www.captureonthego.com/en/promotion/>.
 - **Password:** The password to the above Nu-Book Store.
 - **Recipient(s):** The user name(s) that should receive the document. One or more emails and/or user groups, separated by a comma.
 - **Category:** The category under which the document appears. If the category does not exist, it will be created on the server.

- **Send as Blank Form:** Check this option to send the template as a reusable form. The form will not be deleted from the app's Library when it is submitted. To manually delete it from the Library, swipe it to the left.
- **Document Information group:**
 - **Title:** The title that appears both on the Nu-Book management interface (<https://config-us.captureonthego.com>), as well as on the Capture OnTheGo application on the mobile device. Defaults to the name of the template and the currently active section.
 - **Author:** The name of the author or company.
 - **Description:** The title that appears both on the Nu-Book management interface (<https://config-us.captureonthego.com>), as well as on the Capture OnTheGo application on the mobile device when looking at the document's details.

Send (Test) Email

The Send Email dialog is used to generate mail output and send it to each recipient in the Record Set. To open this dialog, select **File > Send Email**, on the menu.

Note that the subject, recipients etc. must be specified before sending the email; see "Email header settings" on page 373.

For more information about the process of sending out email and the possible settings, see "Generating Email output" on page 973.

Options for this dialog:

- **From** group:
 - **Name:** Enter the name that should appear when sending emails. The name is optional.
 - **Email:** Enter the email address that will appear as a Sender to the email recipient. A single email address should be written.
- **To** group - only when sending a test email:
 - **Email address(es):** Enter one or more email addresses where the test emails are sent. Multiple emails can be separated by semicolons (;), and can be in the same format as above. Note that every email here will receive all the emails for the record-range below.
 - **Use Litmus:** Check to also send the emails to the Litmus test email set in the Email Preferences (to go to the Email Preferences, select **Window > Preferences**, click

the arrow next to **Email**, and then click **General**). Disabled if no Litmus email is set. Also see this how-to: [Test your emails with Litmus](#).

- **Records** group:
 - Select **All**, or click **Selection** and enter the range of records that should be sent. Removing the range disables the selection and sends emails to all records in the record set.
- **Attachments**:
 - **Print context as PDF**: If a Print Context exists in the template, its output will be generated and a PDF version of it will be attached to the outgoing email.
 - **Web Page context as HTML**: If a Web Page Context exists in the template, its output will be generated as a single HTML file with all required resources embedded in the file. This HTML file is then added as an attachment to the outgoing email.
- **Outgoing mail settings**:
 - **Presets**: Use the drop-down to select a preset. These presets are configured in the Email (SMTP) preferences; see "Email SMTP settings" on page 375.

Note

It is recommended to use an Email Service Provider to get access to tools that give you full control over your mailings, like open rates, click through rates etc. See "Using an ESP with PlanetPress Connect" on page 976.

- **Host**: The SMTP server through which the emails are to be sent. Can be a host (mail.domain.com) or an IP address. You can specify a port number as part of the host name, for example: smtp.mandrillapp.com:465.
- **Use authentication**: Check if a username and password are needed to send emails through the host.
- **Start TLS**: Enabled if authentication is checked. Sends emails through Transport Layer Security (TLS), which is sometimes referred to as SSL.
- **User**: Enter the username used to connect to the SMTP server.
- **Password**: Enter the password for the above user name.

Send to Workflow/Files dialog

The Send to Workflow dialog sends templates, data mapping configurations and print presets to the PlanetPress Workflow server, or saves it as a package file. Package files can be sent to other users of the Connect Designer. They cannot be loaded from PlanetPress Workflow.

- **Files to Package** group:
 - **Template:** Select the template to send. By default the currently active template is listed. Click Browse to select another template. As of version 1.3 you may select more than one template in the Browse dialog, and each of them is sent to Workflow or added to a package file.
 - **Data mapping configuration:** Select the data mapping configuration to send. By default the current configuration is listed. Click **Browse** to select another configuration. You may select more than one configuration file in the Browse dialog, and each of them is sent to Workflow or added to a package file.
 - **Job Creation Preset:** Use the drop-down to select a Job Creation Preset to send. Click **Browse** to select a preset that is not in the default save location.
 - **Output Creation Preset:** Use the drop-down to select an Output Creation Preset. Click **Browse** to select a preset that is not in the default save location.
- **Destination** group:
 - **Send files to:** Use the drop-down to select where to send the files.
 - **Workflow machines:** Send the files to a PlanetPress Workflow installation. This lists all the detected PlanetPress Workflow installations detected on the network.
 - **File...:** Click to save the files as a package. This package can be loaded within the Workflow tool.

Select Image dialog

The Select Image dialog lets you select an image, depending on where the image is located.

Resources: lists the images that are present in the Images folder on the Resources pane. A preview of the selected image will be shown at the right.

Disk: lets you select an image file that resides in a folder on a hard drive that is accessible from your computer. A preview of the selected image will be shown below.

- **Path.** The complete syntax of the path is: file://<host>/<path>. Note: if the host is "localhost", it can be omitted, resulting in file:///<path>, for example: file:///c:/resources/images/image.jpg.
- **Browse:** opens an explorer window to browse folders and select an image.

Url lets you select an image file from a specific web address. Select the protocol and then enter the URL (for example, http://www.mysite.com/images/image.jpg). A preview of the selected image will be shown below.

- **Protocol:** http or https.

The option **Save with template**, available when choosing an image from disk or by URL, inserts the file in the **Images** folder on the **Resources** pane. If not saved with the template, the image will remain external.

Note

External images need to be available when the template is merged with a record set to generate output, and their location should be accessible from the machine on which the template's output is produced. External images are updated (retrieved) at the time the output is generated.

Sheet Configuration dialog

The Sheet Configuration dialog defines how different Print context sections output on different Media (see "Media" on page 353) and using different Master Pages (see "Master Pages" on page 350).

General options

The first option defines **Duplex** printing, which also enables or disables the settings for the **Back** side of each sheet.

If Duplex is enabled, you can also:

- Check **Omit empty back side for Last or Single sheet** to reset a page to Simplex if it has an empty back side. Thus changing a Duplex job into a Mixplex job may reduce volume printing costs as omitted back sides aren't included in the number of printed pages.

- Check **Tumble** to duplex pages as in a calendar. (On Portrait output, this would be equivalent to short-edge duplex.)
- Check **Facing pages** to have the margins of the section switch alternately, so that pages are printed as if in a magazine or book.

The Media rotation setting rotates the Media

Sheet position options

With the option **Same for all positions** checked, the same Master Page and Media will be applied to every page in the Print section.

When this option isn't checked, there are multiple groups, each defining the settings for pages grouped by their position within the section as it outputs: **First**, **Middle**, **Last** and **Single** sheets.

Each group defines:

- **Allow content on:** Selects on which face of the sheet content is allowed. If **Front only** or **Back only** is selected, the page acts as a Simplex page even though Duplex printing is enabled. The other page may contain a Master Page, but no contents will be printed on it. As such it does not count in the Content Page Number and Content Page Count markers which can be inserted via the Insert menu (see "Page numbers " on page 345).
- **Media:** Defines the media that is used. If the Media has Virtual Stationery defined, the selected image is shown as a background to each page that corresponds to the media's sheet position.
 - **Edit Script:** Click this button to create a script that defines which Media is used.
- **Master Page Front:** Defines the Master Page used for the front of the selected sheet's position. (Disabled if Back only is selected under Allow content on).
- **Master Page Back:** Defines the Master Page used for the back of the selected sheet's position. (Disabled if Front only is selected under Allow content on, or if Duplex is unchecked.)

Style sheets dialog

The Stylesheet editor dialog is used to edit CSS style sheet resources. For information on the use of style sheets, see "Styling and formatting" on page 551 and "Styling templates with CSS files" on page 553.

This dialog lets you edit the Global style sheet (context_all_styles.css, which by default applies

to all contexts), and the style sheet that applies to the context that is currently being edited in the workspace: Print (context_print_style.css), Email (context_email_style.css) or Web (context_web_style.css).

To open this dialog, select **Edit > Stylesheets....**

- **Context:** Use the drop-down to select **Global** (all contexts) or the context that is open in the workspace, such as **Print**. Selecting a context shows all its CSS rules in the **Rule List**.
- **Show:** Use the drop-down to select whether to show all CSS rules or limit to certain types: **Class**, **ID** or **Element** rules.
- **Rule List:** Displays the list of rules in the currently selected style sheet.
- **Rule Display:** Displays the contents of the currently selected rule in the **Rule List**.
- **New:** Click to create a new rule with the Edit Rule dialog. See "New/Edit Rule dialog" below.
- **Edit:** Click to edit the currently selected rule in the *Rule List* using the Edit Rule dialog. See "New/Edit Rule dialog" below.
- **Delete:** Click to delete the currently selected rule in the *Rule List*.
- **Duplicate:** Click to create a duplicate of the currently selected rule in the *Rule List* using the Edit Rule dialog. The default name for the new rule is the name of the current one plus "-duplicated". See "New/Edit Rule dialog" below.
- **Move Up:** Move the currently selected rule in the *Rule List* up one position in the list.
- **Move Down:** Move the currently selected rule in the *Rule List* down one position in the list.
- **Save:** Click to save all changes to the stylesheet and close the dialog.
- **Cancel:** Click to close the dialog without saving any changes.

New/Edit Rule dialog

The New/Edit Rule dialog shows the properties for a specific CSS selector and how it affects all elements subject to that selector.

At any point you can click on the **Advanced** button to see the Advanced Stylesheet Rule. See "Advanced Stylesheet Rule" on page 731.

- **Name:** The CSS Selector to which this rule applies. Since CSS selectors are not specific to PlanetPress Connect, any selector used in regular CSS can also be used here. See [CSS Selectors on W3Schools](#) for a simple reference page.

Type Tab

- **General group:**
 - **Font:** Select the font used to display text. See also: "Fonts" on page 587. This is equivalent to the CSS `font-family` property.
 - **Size:** Enter the size in a measure, named size or percentage. This is equivalent to the CSS `font-size` property.
 - **Color:** This the color of the text. Select a named font color as defined in the Edit Colors dialog (see "Colors" on page 583) or click the colored square to create a new color or to enter a color manually. The color value must be a valid HTML color name or hexadecimal color code. This is equivalent to the CSS `color` property.
 - **Background Color:** This is the background color of the text. Select a named font color as defined in the Edit Colors dialog (see "Colors" on page 583) or click the colored square to create a new color or to manually enter a color value (a valid HTML color name or hexadecimal color code). This is equivalent to the CSS `background-color` property.
- **Spacing group:**
 - **Letter Spacing:** Set the space between characters in a text in measure or percentage. This is equivalent to the CSS `letter-spacing` property.
 - **Word Spacing:** Set the space between each word in a text in measure or percentage. This is equivalent to the CSS `word-spacing` property.
 - **Whitespace:** Specify how to handle white spaces inside of an element. See [CSS White-Space](#) for details. This is equivalent to the CSS `white-space` property.
- **Style group:** Check any option to apply the selected style to text within the element. This list shows the CSS property and value for each of the options.
 - **Bold:** Sets the `font-weight` to `700`.
 - **Italic:** Sets `font-style` to `italic`.
 - **Underline:** Sets `text-decoration` to `underline`.
 - **Strikethrough:** Sets `text-decoration` to `line-through`.
 - **Subscript:** Sets `vertical-align` to `super`.

- **Superscript:** Sets `vertical-align` to `sub`.
- **Capitalize:** Sets `text-transform` to `capitalize`.
- **Uppercase:** Sets `text-transform` to `uppercase`.
- **Lowercase:** Sets `text-transform` to `lowercase`.
- **Small-caps:** Sets `font-variant` to `small-caps`.

Formats Tab

- **General group:**
 - **Line-height:** Specify the height of each line in the element's text, in measure or percentage. Note that this is not spacing between lines, but rather the complete height of the line itself including the text. Equivalent to the `line-height` property.
 - **Align:** Select how text should be aligned, such as `left`, `center`, `right` or `justify`. Equivalent to the `align` property.
 - **First Indent:** Specify the indentation of the first line of each paragraph in the element. Equivalent to the `text-indent` property.
 - **Display:** Select how to display the element. This can also be used to hide an element completely using the `none` option. See [CSS Display](#). Equivalent to the `display` property.
- **Breaks group:**
 - **Before:** Specifies whether a page break should occur before the element. Equivalent to the `page-break-before` property.
 - **Inside:** Specifies whether to accept page breaks within the element. Equivalent to the `page-break-inside` property.
 - **After:** Specifies whether a page break should occur after the element. Equivalent to the `page-break-after` property.
 - **Widows:** Specifies how to handle widows within a paragraph (lines appearing alone on the next page if the paragraph does not fit on the current one). Equivalent to the `widows` property. Widows and orphans are ignored if the `page-break-inside` property is set to `avoid`.
 - **Orphans:** Specifies how to handle orphans within a paragraph (lines appearing alone at the end of a page if the paragraph does not fit on the current one). Equivalent to the `orphans` property.

Spacing Tab

See also: "Spacing" on page 591.

- **Padding group:** Defines padding (spacing inside the element) in measure or percentage:
 - **All sides:** Check to set all padding to use the Top value. Equivalent to the `border` property.
 - **Top, Left, Bottom, Right:** Set padding for each side. Equivalent to the `border-left`, `border-top`, `border-right` and `border-bottom` properties.
- **Margin group:** Defines margins (spacing outside the element) in measure or percentage:
 - **All sides:** Check to set all margins to use the Top value. Equivalent to the `margin` property.
 - **Top, Left, Bottom, Right:** Set the margin for each side. Equivalent to the `margin-left`, `margin-top`, `margin-right` and `margin-bottom` properties.

Border Tab

See also: "Border" on page 580.

- **Same for all sides:** Defines the border properties for all sides using the Top properties. Equivalent to the `border` property.
- **Top, Left, Bottom, Right:** Each group defines the following properties:
 - **Width:** Specify the thickness of the border. Equivalent to the `border-width` property.
 - **Style:** Specify the style of the border such as `solid`, `dashed` or `dotted`. Equivalent to the `border-style` property.
 - **Color:** Specify the color of the border. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#). Equivalent to the `border-color` property.

Advanced Stylesheet Rule

The Advanced editor is used to manually input rules. Note that to use this dialog, basic knowledge of CSS rules is a pre-requisite, as no check is currently done to verify that properties and values are correct.

- **Property List:** Lists all the currently available properties for the selector.
 - **Property:** The name of the property. This must correspond exactly to a known property (see [CSS Reference](#)). An autocompletion drop-down displays to show possible values when typing.
 - **Value:** The value for the given property. The values must be valid for that property, see the CSS Reference link above and check the property for valid values.
- **New:** Click to create a new line and type in the property.
- **Delete:** Click to delete the currently selected property in the *Property List*.
- **Move Up:** Move the currently selected property in the *Property List* up one position in the list.
- **Move Down:** Move the currently selected property in the *Property List* down one position in the list.

Table Formatting dialog

The Table Formatting dialog defines how a table looks. Note that the settings are applied to the table as a whole. For example, when you change the border of the table, the borders of cells inside the table will not be changed. For more information see "Styling a table" on page 571.

All settings in this dialog are in fact CSS properties. Cascading Style Sheets (CSS) were originally designed for use with web pages: HTML files. Since Designer templates are HTML files, they are styled with CSS. To learn how to use CSS in the Designer, see "Styling and formatting" on page 551 and "Styling templates with CSS files" on page 553. For information about specific properties and their options, see [W3Schools CSS Reference](#).

Table Tab

- **General group:**
 - **Width:** Set the width of the table in measure or percentage. Equivalent to the CSS `width` property.
 - **Height:** Set the height of the table in measure or percentage. Equivalent to the CSS `height` property.
 - **Angle:** Set the rotation angle of the table in clockwise degrees. Equivalent to the CSS `transform: rotate` property.

- **Corner radius:** Set the radius of rounded border corners in measure or percentage. Equivalent to the CSS `border-radius` property.
- **Display:** Use the drop-down or type in the value for how to display the table. Equivalent to the CSS `display` property.
- **Overflow:** Use the drop-down or type in the value for how to handle overflow (text that does not fit in the current size of the box). Equivalent to the CSS `overflow` property.
- **Text wrap group:**
 - **Float:** Use the drop-down or type in the value for how to float the table, if the table is not in an absolute position. Equivalent to the CSS `float` property.
 - **Clear:** Use the drop-down or type in the value for clearing pre-existing alignments. Equivalent to the CSS `clear` property.
- **Positioning:**
 - **Position:** Use the drop-down or type in the value for the type of positioning for the table. Equivalent to the CSS `position` property.
 - **Top:** Set the vertical offset between this table and its parent's top position. Equivalent to the CSS `top` property.
 - **Left:** Set the horizontal offset between this table and its parent's left position. Equivalent to the CSS `left` property.
 - **Bottom:** Set the vertical offset between this table and its parent's bottom position. Equivalent to the CSS `bottom` property.
 - **Right:** Set the horizontal offset between this table and its parent's left position. Equivalent to the CSS `right` property.
 - **Z-index:** Set the z-index of the table. The z-index defines in which order elements appear. Equivalent to the CSS `z-index` property.
- **Breaks group:**
 - **Before:** Specifies how to handle page breaks before the table. Equivalent to the CSS `page-break-before` property.
 - **Inside:** Specifies whether to accept page breaks within the table. Equivalent to the CSS `page-break-inside` property.
 - **After:** Specifies how to handle page breaks after the table. Equivalent to the CSS `page-break-after` property.

- **Widows:** Specifies how to handle widows within the table (rows appearing alone on the next page if the table does not fit on the current one). Equivalent to the CSS `widows` property. Widows and orphans are ignored if the `page-break-inside` property is set to `avoid`.
- **Orphans:** Specifies how to handle orphans within the tables (rows appearing alone at the end of a page if the table does not fit on the current one). Equivalent to the CSS `orphans` property.

Spacing Tab

For information about spacing see "Spacing" on page 591.

- **Padding group:** Defines padding (spacing inside the element) in measure or percentage:
 - **All sides:** Check to set all padding to use the Top value. Equivalent to the CSS `padding` property.
 - **Top, Left, Bottom, Right:** Set padding for each side. Equivalent to the CSS `padding-left`, `padding-top`, `padding-right` and `padding-bottom` properties.
- **Margin group:** Defines margins (spacing outside the element) in measure or percentage:
 - **All sides:** Check to set all margins to use the Top value. Equivalent to the CSS `margin` property.
 - **Top, Left, Bottom, Right:** Set the margin for each side. Equivalent to the CSS `margin-left`, `margin-top`, `margin-right` and `margin-bottom` properties.

Border Tab

For information about borders see "Border" on page 580.

- **Same for all sides:** Defines the border properties for all sides using the Top properties. Equivalent to the CSS `border` property.
- **Top, Left, Bottom, Right:** Each group defines the following properties:
 - **Width:** Specify the thickness of the border. Equivalent to the CSS `border-width` property.

- **Style:** Specify the style of the border such as `solid`, `dashed` or `dotted`. Equivalent to the CSS `border-style` property.
- **Color:** Specify the color of the border. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#). Equivalent to the CSS `border-color` property.

Background Tab

For information about backgrounds see "Background color and/or image" on page 579.

- **General group:**
 - **Color:** Specify the color of the table background. Click the small square to the right to open the Color Picker (see "Color Picker" on page 674). Equivalent to the CSS `background-color` property.
- **Background image group:**
 - **Source:** click the **Select Image** button to select an image via the "Select Image dialog" on page 725. Equivalent to the CSS `background` property.
 - **Size:** select `auto`, `cover` or `contain` (for an explanation see http://www.w3schools.com/cssref/css3_pr_background-size.asp), or type the width and height of the image in a measure (e.g. `80px 60px`) or as a percentage of the parent element's size (e.g. `50% 50%`). Equivalent to the CSS `background-size` property.
 - **Position:** select the position for the background-image. Equivalent to the CSS `background-position` property.

Table Cell Formatting dialog

The Table Cell Formatting dialog defines how a particular cell in a table looks. For more information see "Styling a table" on page 571.

All settings in this dialog are in fact CSS properties. Cascading Style Sheets (CSS) were originally designed for use with web pages: HTML files. Since Designer templates are HTML files, they are styled with CSS. To learn how to use CSS in the Designer, see "Styling and formatting" on page 551 and "Styling templates with CSS files" on page 553. For information about specific properties and their options, see [W3Schools CSS Reference](#).

Cell Tab

- **Width:** Set the width of the table in measure or percentage. Equivalent to the CSS `width` property.
- **Height:** Set the height of the table in measure or percentage. Equivalent to the CSS `height` property.
- **Vertical Align:** Specify how text is vertically aligned in the cell: top, middle, bottom or baseline. With the baseline value all the table data share the same baseline. Often this has the same effect as the bottom value. However, if the fonts are in different sizes, baseline looks better.

Type Tab

- **General group:**
 - **Font:** Select the font used to display text, equivalent to the CSS `font-family` property.
 - **Size:** Enter the size in measure, named size or percentage, equivalent to the CSS `font-size` property.
 - **Color:** Select a named font color as defined in the [Colors Editor](#), create a new color or enter a color manually for text to be displayed. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#). Equivalent to the CSS `color` property.
- **Spacing group:**
 - **Letter Spacing:** Set the space between characters in a text in measure or percentage. Equivalent to the CSS `letter-spacing` property.
 - **Word Spacing:** Set the space between each word in a text in measure or percentage. Equivalent to the CSS `word-spacing` property.
 - **Whitespace:** Specify how to handle white spaces inside of an element. Equivalent to the CSS `white-space` property. See [CSS White-Space](#) for details.
- **Style group:** Check any option to apply the selected style to text within the element:
 - **Bold:** Sets the `font-weight` to 700.
 - **Italic:** Sets the `font-style` to `italic`.
 - **Underline:** Sets the `text-decoration` to `underline`.
 - **Strikethrough:** Sets the `text-decoration` to `line-through`.

- **Subscript:** Sets the `vertical-align` to `super`.
- **Superscript:** Sets the `vertical-align` to `sub`.
- **Capitalize:** Sets the `text-transform` to `capitalize`.
- **Uppercase:** Sets the `text-transform` to `uppercase`.
- **Lowercase:** Sets the `text-transform` to `lowercase`.
- **Small-caps:** Sets the `font-variant` to `small-caps`.

Spacing Tab

For information about spacing see "Spacing" on page 591.

- **Padding group:** Defines padding (spacing inside the element) in measure or percentage:
 - **All sides:** Check to set all padding to use the Top value. Equivalent to the CSS `border` property.
 - **Top, Left, Bottom, Right:** Set padding for each side. Equivalent to the CSS `padding-left`, `padding-top`, `padding-right` and `padding-bottom` properties.

Border Tab

For information about borders see "Border" on page 580.

- **Same for all sides:** Defines the border properties for all sides using the Top properties. Equivalent to the `border` property.
- **Top, Left, Bottom, Right:** Each group defines the following properties:
 - **Width:** Specify the thickness of the border. Equivalent to the CSS `border-width` property.
 - **Style:** Specify the style of the border such as `solid`, `dashed` or `dotted`. Equivalent to the CSS `border-style` property.
 - **Color:** Specify the color of the border. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#). Equivalent to the CSS `border-color` property.

Background Tab

For information about backgrounds see "Background color and/or image" on page 579.

- **General group:**
 - **Color:** Specify the color of the table cell background. The color value must be a valid [HTML Color Name](#), or a valid [HTML Hex Color](#). Equivalent to the CSS `background-color` property.
- **Background image group:**
 - **Source:** click the **Select Image** button to select an image via the "Select Image dialog" on page 725. Equivalent to the CSS `background` property.
 - **Size:** select `auto`, `cover` or `contain` (for an explanation see http://www.w3schools.com/cssref/css3_pr_background-size.asp), or type the width and height of the image in a measure (e.g. `80px 60px`) or as a percentage of the parent element's size (e.g. `50% 50%`). Equivalent to the CSS `background-size` property.
 - **Position:** select the position for the background-image. Equivalent to the CSS `background-position` property.

Keyboard shortcuts

This topic gives an overview of keyboard shortcuts that can be used in the Designer.

Although some of the keyboard shortcuts are the same, this isn't a complete list of Windows keyboard shortcuts. To find that, refer to the documentation of the Windows version that you are using.

Menu items

The following key combinations activate a function in the menu.

Key combination	Function
Alt	Put the focus on the menu. (Alt + the underlined letter in a menu name displays the corresponding menu.) The menu can then be browsed using the Enter key, arrow up and arrow down buttons.
Alt + F4	Exit

Key combination	Function
Alt + Shift + D	Design view
Alt + Shift + E	Preview HTML
Alt + Shift + L	Live view
Alt + Shift + P	Preview view
Alt + Shift + S	Source view
Ctrl + C or: Ctrl + Insert	Copy
Ctrl + F	Find
Ctrl + I	Properties
Ctrl + N	New
Ctrl + O	Open file
Ctrl + P	Print
Ctrl + R	Show/hide rulers
Ctrl + S	Save file
Ctrl + V or: Shift + Insert	Paste
Ctrl + X or: Shift + Delete	Cut
Ctrl + W	Close file

Key combination	Function
Ctrl + Y or: Ctrl + Shift + Z	Redo
Ctrl + Z	Undo
Ctrl + Alt + ;	Lock guides
Ctrl + Shift + R	Clear preview cache
Ctrl + Shift + S	Save all
Ctrl + Shift + W	Close all
Ctrl + Shift + ;	Snap to guides
Ctrl + ;	Show guides
Ctrl + '	Show/hide virtual stationery
Ctrl + \	Highlight master page items
Ctrl + F5	Revert
Ctrl + F10	Save as
Ctrl + F11	Send COTG test
Ctrl + F12	Send to Workflow / Package files

Workspace

The following key combinations activate a function in the Workspace.

Key combination	Function
Alt + -	Open system menu
Alt + F7 or: Alt + Page Down	Next tab
Alt + Shift + F7 or: Alt + Page Up	Previous tab
Ctrl + A	Select all
Ctrl + Shift + E	Switch to Editor
Ctrl + F	Find (opens the Find and Replace dialog: "Find/Replace Dialog" on page 677)
Ctrl + F6	Next editor (when there is more than one file open in the Workspace)
Ctrl + Shift + F6	Previous editor (when there is more than one file open in the Workspace)
Shift + F10 or: Ctrl + Shift + F10	Open context menu

Design and Preview tab

In addition to the keyboard shortcuts for menu items and the Workspace, the following key combinations have a special function in the **Design** tab and **Preview** tab of the Workspace.

Key combination	Function
Ctrl + B	Bold (works on a text selection)
Ctrl + E	Open Paragraph formatting dialog

Key combination	Function
Ctrl + I	Italic (works on a text selection)
Ctrl + H	Show Edges
Ctrl + K	Delete browser element
Ctrl + M	Open Box formatting dialog
Ctrl + T	Open Text formatting dialog
Ctrl + U	Underline ((works on a text selection)
Ctrl + + or: Ctrl + Shift + + or: Ctrl + = or: Ctrl + Shift + =	Zoom in
Ctrl + - or: Ctrl + Shift + -	Zoom out
Ctrl + 0	Zoom to page width
Ctrl + 1	Zoom to page content width
Ctrl + arrow up	Select parent
Ctrl + Shift + R or: F5	Refresh

Text editors: Source tab, JavaScript, CSS, Script Editor

The following key combinations have a special function in the **Source** tab of the Workspace (see also: "Source tab" on page 770), and when editing a JavaScript or CSS file in the Workspace, and in the Script Editor (expanded view).

Key combination	Function
Ctrl + space	Content assist (auto-complete)
Ctrl + A	Select all
Ctrl + D	Duplicate line
Ctrl + F	Find
Ctrl + I	Indent (Tab)
Ctrl + J	<ul style="list-style-type: none"> • Script Editor: Add a line break. • Other editors: Incremental find; start typing a search string directly after pressing this key combination.
Ctrl + K	Find next
Ctrl + L	Go to line; a prompt opens to enter a line number.
Ctrl + Shift + D	Delete line
Tab	Expand Emmet abbreviation (see "Emmet Preferences" on page 706)
Shift + Tab	Shift selected lines left
Tab	Shift selected lines right
Ctrl + /	Comment out / uncomment a line in code
Ctrl + Shift + /	Comment out / uncomment a code block

Scripts pane and Resources pane

The following keys or key combinations have a special function when a file is selected in the **Resources** pane and when a script is selected in the **Scripts** pane.

Key combination	Function
F2	Rename
Alt + Enter	Open Properties dialog
Delete	Delete

Data Model pane

You can use the following keys to browse records in the Data Model pane:

- Page Up: next record
- Page Down: previous record
- Home: first record
- End: last record.

Menus

The following menu items are shown in the Designer menu. For a list of keyboard shortcuts, see "Keyboard shortcuts" on page 738.

File Menu

- **New...:** Opens the **New (Select a Wizard)** dialog. You can choose from the Email, Print or Web Template Wizards. See "Templates" on page 304.
- **Open:** Opens a standard File Open dialog. This dialog can be used to open Templates and Data Mapping Configurations. See "Templates" on page 304 and "Data mapping configurations" on page 101.
- **Open Recent:** Lists the most recently opened Templates and configurations. Clicking on a template will open it in the Designer module, clicking on a Data Mapping Configuration will open it in the DataMapper module.
- **Close:** Closes the currently active Data mapping configuration or Template. If the file needs to be saved, the appropriate Save dialog will open.

- **Close All:** Closes any open Data Mapping Configuration or template. If any of the files needs to be saved, the Save Resources dialog opens.
- **Close Others:** Closes all Data mapping configuration and templates except the one that is currently active in the workspace.
- **Save:** Saves the current Data mapping configuration or Template to its current location on disk. If the file has never been saved, the Save As dialog appears instead.
- **Save All:** Saves all open files. If any of the open files have never been saved, the Save As dialog opens for each new unsaved file.
- **Save As...:** Saves the current file to a new location on disk.
- **Revert:** Appears only in the Designer module. Reverts all changes to the state in which the file was opened or created.
- **Add Data:** Adds data either to the current data mapping configuration or to the open template. See "Loading data" on page 594 .
 - **From File Data Source...:** Opens the dialog to add a new data file to the currently loaded data mapping configuration. Not available if the currently loaded data mapping configuration connects to a database source.
 - **From Database Data Source...:** Opens the Edit Database Configuration dialog. Not available if the currently loaded data mapping configuration is file-based.
 - **Generate Counters:** Opens the Generate Counter Wizard to create a custom counter as a data source.
- **Send to Workflow...:** Opens the "Send to Workflow/Files dialog" on page 725 to send files to a local Workflow software installation.
- **Export report:** opens the wizard to save a template report. See "Exporting a template report" on page 308.
- **Properties:** opens the File Properties dialog.
- **Print:** Opens the "Print Options" on page 851 dialog.
- **Print Presets:** Selecting this option allows you to create or modify Printing Presets, which can be saved and used in print runs thereafter.
 - **Job Creation Presets:** Opens the "Job Creation Presets" on page 840 dialog.
 - **Output Creation Presets:** Opens the "Output Creation Settings" on page 850 dialog.

- **Proof Print:** Opens the "Print Options" on page 851 dialog as a Proof Print dialog which limits the number of records output. The options themselves are identical to the regular Print Output dialog.
- **Send Email:** Opens the Send Email dialog; see "Send (Test) Email" on page 723 and "Generating Email output" on page 973.
- **Send Test Email:** Opens the Send Test Email dialog; see "Send (Test) Email" on page 723.
- **Send COTG Test:** Opens the "Send COTG Test" on page 722 dialog, to send the current Web Context to the Capture OnTheGo Application.
- **Exit:** Closes the software. If any of the files needs to be saved, the Save Resources dialog opens.

Edit Menu

- **Undo <action>:** Undoes the previous action that was done.
- **Redo <action>:** Redoes an action that was previously undone.
- **Cut:** Cuts the currently selected text, object or element and puts it on the clipboard.
- **Copy:** Copies the the currently selected text, object or element to the clipboard.
- **Copy to snippet:** Creates a new snippet from the selected text, object or element.
- **Paste:** Takes the current clipboard content and pastes it at the pointer location.
- **Delete Browser Element:** Removes the currently selected element in the workspace.
- **Find/Replace:** Only active while inside the Workspace. Opens the [Find/Replace](#) dialog.
- **Stylesheets...:** Open the "Style sheets dialog" on page 727. See "Styling and formatting" on page 551 and "Styling templates with CSS files" on page 553.
- **Colors...:** Opens the [Colors Editor](#) dialog. See "Colors" on page 583.
- **Fonts...:** opens the "Font Manager" on page 679. See "Fonts" on page 587.
- **Locale...:** Opens the [Locale Settings](#) dialog. See "Locale" on page 590.
- **Color Settings...:** Opens the [Color Settings](#) dialog. See "Colors" on page 583.

Insert Menu

- **Image:** Inserts an image using a resource that is local to the template, a resource on disk or a URL. See "Images" on page 537.

- **Text:**
 - **Wrap in span:** Wraps selected text in a element. The ID or class of the span can be used as a selector for scripts and styles.
- **Special Characters:** Displays a categorized list of special HTML characters that can be inserted at the current pointer location. When a character is clicked, its HTML Entity is inserted. This includes:
 - **Symbols:** Use the list to insert a special symbol such as Copyright, Trademark, or Ellipsis.
 - **Markers:** Use the list to insert pagination markers that are replaced with specific page numbering:
 - **Page Number:** This marker is replaced by the current page number in the document. Even if the page number is not used on certain pages, those page are still added to the page count.
 - **Page Count:** This marker is replaced by the total number of pages in the document, including pages with no contents.
 - **Content Page Number:** This marker is replaced by the current page number (with contents) in the document.
 - **Content Page Count:** This marker is replaced by the total number of pages that have contents in them, in the document. A page with contents is a page that is part of a section that has variable data on it. A page with a Master Page but no contents (set in the Sheet Configuration tab of the "Print section properties" on page 720) is not included in the Content Page Count.
 - **Sheet Number:** This marker is replaced by the current sheet number (physical piece of paper with two sides, or pages) in the document. This is equivalent to half the page number, for example if there are 10 pages, there will be 5 sheets.
 - **Sheet Count:** This marker is replaced by the total number of sheets in the document, whether or not they have contents.
 - **Dashes and Spaces:** Use the list to insert special dashes, such as an em-dash, and spaces, such as non-breaking spaces or an en-space. (The HTML code inserted for the dash or space is visible on the Source tab of the workspace.)
 - **Arrows:** Use the list to insert directional arrows (in one of four directions).
 - **Geometric Shapes:** Use the list to insert a special geometric shape, such as circles, triangles and squares.

- **Date:** Opens the "Date" on page 526 dialog to add a date to the template based on the current system's date and time.
- **Wrap in box:** Puts the element in which the cursor is located in an inline box (a <div>).
- **Table**
 - **Thead, tbody, tfoot:** Insert a header, body or footer (if not already present) in the current table.
 - **Standard:** Inserts a table with a specific number of columns and rows through the Standard Table Wizard; see "Table" on page 542.
 - **Dynamic:** Inserts a dynamic table where the number of rows is determined by a Details table, through the Dynamic Table Wizard; see "Dynamic table" on page 618.
- **Table Elements:**
 - **Insert Row Above:** Inserts a row above the current one. The row configuration, such as merged cells and cell styles, is duplicated, but contents is not.
 - **Insert Row Below:** Inserts a row below the current one. The row configuration, such as merged cells and cell styles, is duplicated, but contents is not.
 - **Insert Column Before:** Inserts a column to the left of the current one. The column configuration, such as merged cells and cell styles, is duplicated, but contents is not.
 - **Insert Column After:** Inserts a column to the right of the current one. The column configuration, such as merged cells and cell styles, is duplicated, but contents is not.
- **Common Elements:**
 - **Paragraph...:** Opens a dialog to add a <p> element; see "Text and special characters" on page 546.
 - **H1 through H6...:** Opens a dialog to add a <h1> to <h6> element; see "Text and special characters" on page 546.
 - **Address...:** Opens a dialog to add an <address> element.
 - **Preformatted...:** Opens a dialog to add a <pre> element.
- **Structural Elements:**
 - **Div...:** Opens a dialog to add a <div> element; see "Boxes" on page 513
 - **Span...:** Opens a dialog to add a element; see "Boxes" on page 513
 - **Article...:** Opens a dialog to add an <article> element

- **Section...**: Opens a dialog to add a <section> element (the HTML element, not a section in a context).
- **Header...**: Opens a dialog to add a <header> element.
- **Footer...**: Opens a dialog to add a <footer> element.
- **Nav...**: Opens a dialog to add a <nav> element.
- **Aside...**: Opens a dialog to add an <aside> element.

Note

Article, Section, Header, Footer, Nav and Aside are HTML5 semantic elements; see http://www.w3schools.com/html/html5_semantic_elements.asp

- **Form Elements** (see "Form Elements" on page 532)
 - **Form...**: Opens a dialog to add a Form Element; see "Forms" on page 527.
 - **Fieldset...**: Opens a dialog to add a Fieldset Element; see "Fieldset" on page 533.
 - **Text Field...**: Opens a dialog to add a Text Field; see "Text" on page 533.
 - **Email Field...**: Opens a dialog to add an Email Field; see "Email" on page 533.
 - **URL Field...**: Opens a dialog to add a URL Field; see "URL" on page 533.
 - **Password Field...**: Opens a dialog to add a Password Field; see "Password" on page 533.
 - **Text Area...**: Opens a dialog to add a Text Area; see "Text area" on page 533.
 - **Date Field...**: Opens a dialog to add a Date Field; see "Date" on page 533.
 - **Number Field...**: Opens a dialog to add a Number Field; see "Number" on page 533.
 - **Hidden Field...**: Opens a dialog to add a Hidden Field; see "Hidden field" on page 534.
 - **Label...**: Opens a dialog to add a Label; see "Label" on page 534.
 - **Checkbox Field...**: Opens a dialog to add a Checkbox; see "Checkbox" on page 534.

- **Radio Button...:** Opens a dialog to add a Radio Button; see "Radio Button" on page 534.
- **Select Field...:** Opens a dialog to add a Select (drop-down); see "Select" on page 535.
- **Button...:** Opens a dialog to add a Button; see "Button" on page 535.
- **Help text:** Opens a dialog to insert a paragraph (<p>) for help text.
- **COTG Form Elements** (see "COTG Elements" on page 519).
 - **Signature...:** Opens a dialog to add a Signature element, see "Signature" on page 525.
 - **Date...:** Opens a dialog to add a Date Element, see "Date and Formatted Date" on page 523.
 - **Date Formatted...:** Opens a dialog to add a Formatted Date Element, see "Date and Formatted Date" on page 523.
 - **Time...:** Opens a dialog to add a Time element, see "Time and Formatted Time" on page 525.
 - **Time Formatted...:** Opens a dialog to add a Formatted Time element, see "Time and Formatted Time" on page 525.
 - **Geolocation...:** Opens a dialog to add a Geolocation element, see "Geolocation" on page 524.
 - **Locale...:** Opens a dialog to add a Locale element, see "Locale" on page 525.
 - **Camera...:** Opens a dialog to add a Camera element, see "Camera" on page 520.
 - **Image and annotation:** Opens a dialog to add an image that can be annotated by the user; see "Image & Annotation" on page 524.
 - **Barcode Scanner...:** Opens a dialog to add a Barcode Scanner element, see "Barcode Scanner" on page 519.
 - **User Account...:** Opens a dialog to add a User Account element, see "User Account" on page 526.
 - **Device Info...:** Opens a dialog to add a Device Info element, see "Device Info" on page 523.
 - **Repository ID:** Opens a dialog to add a Repository ID element, see "Repository ID" on page 525.

- **Document ID:** Opens a dialog to add a Document ID element, see "Document ID" on page 523.
- **Fields Table:** Opens a dialog to add a Fields Table element, see "Fields Table" on page 524.
- **Form Wizard:** Opens the Form Wizard to add a form to a Web context; see "Forms" on page 527
- **Validation Wizard:** Opens the Validation Settings dialog to change the validation settings on the currently selecting tools; see "Changing a Form's validation method" on page 531.
- **Business Graphic:** Displays a list of available business graphic object to be inserted:
 - **Insert Pie Chart:** Opens the Pie Chart script dialog to insert a new Pie Chart.
 - **Insert Bar Chart:** Opens the Bar Chart script dialog to insert a new Bar Chart.
 - **Insert Line Chart:** Opens the Line Chart script dialog to insert a new Line Chart.
- **Barcode:** Displays a list of available barcodes. Click on one to insert it in the page. See "Barcode" on page 470.

Format Menu

- **Size:** When text is selected, choose a predefined or custom font size in this submenu to change the size of the selected text.
 - **Other...:** Opens the Text Formatting dialog for advanced style selection; see "Styling text and paragraphs" on page 562.
 - **7pt - 72pt:** Sets the size of the selected text to the chosen font size.
- **Style:** When text is selected, sets the text style by applying or removing the following attributes: Plain, Bold, Italic, Underline, Strikethrough, Subscript, Superscript, Capitalize, Uppercase, Lowercase, Small-caps. This is the same as opening the Text Formatting dialog (**Format > Text**) and checking the appropriate style. See "Styling text and paragraphs" on page 562.
- **Color:** When text is selected, sets the text color by applying the color attribute to the text. The color submenu lists all the colors in the [Colors Editor](#).
- **Text...:** Opens the Text Formatting dialog to modify the current text selection. See "Styling text and paragraphs" on page 562.
- **Align:** When an element is selected, determines how its contents is aligned inside the element. Options are Align Left, Align Right, Align Center and Justify.

- **Paragraph...:** Opens the "Paragraph Formatting dialog" on page 689 to modify the paragraph where the cursor is located. See "Styling text and paragraphs" on page 562.
- **Paragraph Format:** Displays a list of generic element types that can be used for a text element. Selecting one of them converts the element where the cursor is located into the appropriate element (for example `<p>` for Paragraph, `<h3>` for Heading 3, etc).
- **Float**
 - **Left:** Floats the current element to the left. This is equivalent to setting the CSS `float` property to `left`.
 - **Right:** Floats the current element to the right. This is equivalent to setting the CSS `float` property to `right`.
 - **None:** Removes any float style applied to the currently selected element.
- **Box...:** Opens the "Box Formatting dialog" on page 670 to modify the box where the cursor is located.
- **Image...:** Opens the "Image Formatting dialog" on page 680 to modify the image that is currently selected.
- **Table...:** Opens the "Table Formatting dialog" on page 732 to modify the table in which the cursor is located. If the cursor is within a table embedded within another, the innermost table's formatting is the one modified.
- **Table Cell...:** Opens "Table Cell Formatting dialog" on page 735 to modify the cell where the cursor is located.
- **Hyperlink**
 - **Insert...:** Creates a hyperlink on the currently selected text or element and opens its properties; see "Hyperlink and mailto link" on page 536.
 - **Edit...:** Opens the properties for the currently selected hyperlink; see "Hyperlink and mailto link" on page 536.
 - **Remove:** Removes the currently selected hyperlink. The text or element that was the hyperlink is not removed.

Context Menu

- **Add:**
 - **Print Context:** Adds a new Print context to the template if one does not exist.
 - **HTML Email Context:** Adds a new Email context to the template if one does not

exist.

- **Web Page Context:** Adds a new Web context to the template if one does not exist.
- **Delete:** Deletes the currently selected context. The last remaining context cannot be deleted.
- **Go to:** Opens the first section in the selected context. This is the same as double-clicking on the first section of any context in the Resource Pane.
- **Finishing:** Opens the Print context's Finishing options dialog (see "Setting the binding style for the Print context" on page 334). This option is only available when editing a Print section in the Workspace.
- **PDF Attachments:** Opens a dialog to set the compression for PDF attachments; see "PDF Attachments dialog" on page 692. This option is only available when editing an Email section in the Workspace.
- **Includes:** Opens the Includes dialog; see "Includes dialog" on page 682. This option is only available when editing a Web section in the Workspace.
- **Preview HTML:** Opens the currently selected section in the default system browser to preview it. This feature works in all contexts.
- **Profile Scripts:** Opens the "Profile Scripts dialog" on page 714 to test script performance (see "Testing scripts" on page 632).
- **Preflight:** Opens the Preflight dialog. Preflight verifies the template for common errors (see "Testing scripts" on page 632).

Section Menu

- **Add:** Adds a new section to the currently selected context.
- **Delete:** Deletes the currently selected section.
- **Arrange:** Opens the "Arrange Sections" on page 721 dialog.
- **Go to:** Lists the sections in the currently selected context. Open one by clicking it.
- **Properties...:** Opens the appropriate section properties: Email , Print or Web. See "Section properties dialogs" on page 719.
- **Includes...:** Opens the "Includes dialog" on page 682.
- **Finishing...** (Print Sections only): Opens the Finishing tab in the "Print section properties" on page 720.
- **Sheet Configuration...** (Print Sections Only): Opens the "Sheet Configuration dialog" on page 726.

- **Master Pages:** Lists the available Master Pages in the template (see "Master Pages" on page 350). Open one by clicking it.
- **Master Page Properties...:** Opens the currently selected Master Page's properties dialog; see "Master Pages" on page 350.

View Menu

- **50/75/100/150/200%:** Zooms the [Workspace](#) at the selected level.
- **Source View:** Shows the HTML source for the template, including CSS and HTML code.
- **Design View:** Shows the template including all styles, text and images as well as the placeholders used for variable data.
- **Preview View:** Shows the template as it will output with the current record, with the personalized content (see "Personalizing Content" on page 592).
- **Refresh:** Reloads the view, including static external images and remote stylesheets, and re-runs the scripts (the latter in Preview Mode only).
- **Show Edges:** Shows or hides a colored border around elements on the page.
- **Rulers:** Shows or hides the rulers in the [Workspace](#). Rulers only appear for Print contexts.
- **Guides:**
 - **Show Guides:** Shows or hides the margin lines and guides in a Print section (see "Print" on page 325, "Page settings: size, margins and bleed" on page 344 and "Guides" on page 568). The colors of margin lines and guides are adjustable; see "Editing preferences" on page 704.
 - **Lock Guides:** Locks the guides, so that they cannot accidentally be moved while working on the Print context.
 - **Snap to Guides:** Enables or disables snapping to guides and to margins when moving objects.
- **Virtual Stationery:** Enables or disables the visibility of the PDF Background image set in the Media.
- **Highlight Master Page Items:** Enables or disables a yellow border around Master Page items in a section.
- **Object Resizing:** Enables or disables the ability to resize <div> elements on the page. See "Editing preferences" on page 704 for more fine-tuned control.

Window Menu

- **Show View>**: Use the options in this menu to show or hide different panes of the UI.
 - **Properties > Attributes**: Shows the [Attributes pane](#)
 - **Messages**: Shows the Messages pane, see "Preflight Results and Messages" on page 759.
 - **Problems**: Shows the Problems pane, see "Preflight Results and Messages" on page 759.
 - **Resources**: Shows the [Resources pane](#)
 - **Outline**: Shows the [Outline pane](#)
 - **Data Model**: Shows the [Data Model pane](#)
 - **Scripts**: Shows the [Scripts pane](#)
- **Reset Perspective**: Resets all toolbars and panes to the initial configuration of the module.
- **Preferences**: Click to open the [Preferences](#) dialog.

Help Menu

- **Software Activation**: Displays the Software Activation dialog. See "Activating a License" on page 49.
- **Help Topics**: Opens the help system in the default web browser.
- **Contact Support**: Opens the [Objectif Lune Contact Page](#) in the default system web browser.
- **About PlanetPress Connect Designer**: Displays the software's About dialog.
- **Welcome Screen**: Re-opens the Welcome Screen.

Panes

Panes are windows containing user interface elements (such as information or properties), which can be docked and undocked, moved around and merged together through tabbed panes.

Here is a list of all panes:

Attributes pane

The Attributes pane displays all of the properties of the currently selected object in the Workspace. These properties vary greatly depending on the object that has been selected.

General

These attributes are common to all elements in the template and will always appear.

- **ID:** A unique identifier for the selected element. Used for CSS selections as well as JavaScript expressions affecting single elements.
- **Class:** One or more classes that can be common to more than one elements. Used for CSS selections and JavaScript expressions that can affect multiple elements.

Other

These attributes are available depending on the item selected (in parenthesis).

- **Whitespace element :** Check to make the element a whitespace element, meaning it will only appear on the page if there is enough space for it. This is useful for templates with variable height elements or conditional elements, to fill empty spaces with transpromotional material. Note that only top-level elements (a paragraph not inside a table or a div) will function at whitespace elements.
- **Source (image):** The location of the image file. For image resources in the template, the image path is often `images/<imagefile>.<extension>`
When the source is a PDF, an addition button appears next to this box that opens the "Select Image dialog" on page 725.
- **Alternate text (image):** The "Alt" text used when hovering over the image in a browser. Also used for accessibility.
- **Cell Spacing (table only):** Defines the *cellspacing* attribute of the table which controls the spacing between cells in the table.
- **Cell Padding (table only):** Defines the *cellpadding* attribute of the table which controls the padding inside each cell of the table.
- **Column Resizing (table only):** Check to enable columns to be resized directly within the [Workspace](#).
- **Detail Table (table only):** Defines which detail table the repeat of the table is based on. The number of detail lines in the table is the number of the time the repeating row (see below) is repeated.

- **Title (table only):** Defines the title of the table. This has no impact on the table's displays, only on accessibility of HTML pages and screen readers.
- **Repeat (table row not in <tfoot> or <thead> only):** Defines if the row is affected by the detail table calculation. This row is the one repeated in a Dynamic Table.
- **Show Row (table row only):** Use the drop-down to determine when the selected row appears when a dynamic table overflows. This option is only available in a row manually added inside of a Dynamic Table.
 - **Before page break:** The row will appear on all pages except the last one.
 - **At end of table:** The row will appear only on the last page.
 - **Always:** The row will appear on every page of the table.
- **Subtotal Line (table row inside a <tfoot> only):** Defines the footer row as the place where the SubTotal is displayed. This is the row where a subtotal script is expected to display the result.
- **Type (form input element):** Use the drop-down to select an input type. The drop-down lists all input types, including HTML5 input types (see http://www.w3schools.com/html/html_form_input_types.asp).

Geometry

These attributes are available for certain elements that have position or size attributes such as images and boxes.

- **X-Offset:** The horizontal distance from the top-left of the object to the left position of its parent. This is used only for relative and absolute positioned elements.
- **Y-Offset:** The vertical distance from the top-left of the object to the top position of its parent.
- **Width:** The width of the element, by default in pixels. For an image, this defaults to the original image width in pixels.
- **Height:** The height of the element, by default in pixels. For an image, this defaults to the original image height in pixels.

Note

When no unit is added to a geometry value, the default unit will be added to the value; see "Print Preferences" on page 710.

Page

These attributes appear when selecting the *Page* node in the [Outline Pane](#).

- **Master Page:** Which of the "Master Pages" on page 350 to use for the template.

Data Model pane

The Data Model Pane displays a Data Model used to help design the template, along with (optional) extracted data. When executing a data mapping configuration or directly loading data (see "Loading data" on page 594), the resulting record set is loaded in the Data Model Pane. The information shown is the extracted information for the current record within the record set.

Data is displayed as a tree view, with the root level being the record table, levels below it being detail tables, and any level below being called "nested tables".

Pane options

- **Minimize/Maximize:** Click to minimize or maximize the pane. See [Moving and Merging Panes](#).
- **Import Data Model:** Click to browse to a Data Model File and import it. Importing a Data Model File displays the file's data model structure into the Data Model Pane, with optional sample data for each field.
- **Export Data Model:** Click to browse to a location to save the Data Model File and save it.
- **Synchronize Data Model:** Click to synchronize the data model with the one currently loaded in the open Data Mapping Configuration. Disabled if no configuration is currently open.

Using the Data Model

When a Data Model is loaded inside of the Data Model Pane, it can be used to design templates by dragging the fields directly into the template; see "Variable Data" on page 604. If data is present (from a Data Model File or a Data Mapping Configuration), it is possible to preview the resulting data in the template using the **Preview** tab (see [Workspace](#)).

You can use the following keys to browse records in the Data Model pane:

- Page Up: next record
- Page Down: previous record

- Home: first record
- End: last record.





Preflight Results and Messages

Messages pane

The **Messages** pane is shared between the **DataMapper** and **Designer** modules and displays any warnings and errors from the data mapping configuration or template.

To open it in the Designer module, click the Messages button at the bottom right of the window (see "Designer User Interface" on page 666).

Buttons

- **Export Log** : Click to open a **Save As** dialog where the log file (.log) can be saved on disk.
- **Clear Log Viewer** : Click to remove all entries in the log viewer.
- **Filters** : Displays the [Log Filter](#).
- **Activate on new events** : Click to disable or enable the automatic display of this dialog when a new event is added to the pane.

Field headers

- **Message**: The contents of the message, indicating the actual error.
- **Component**: Whether the entry is a warning or an error.
- **Source**: The source of the error. This indicates the name of the step as defined in its step properties.
- **Date**: The date and time when the error occurred.

Preflight Results pane

The Preflight Results pane displays any notifications or errors related to the template, its scripts, its code or output generation. See also: "Testing scripts" on page 632.

Log Filter

The log filter determines what kind of events are show in the [Messages Pane](#).

- **Event Types** group:
 - **OK**: Uncheck to hide OK-level entries.
 - **Information**: Uncheck to hide information-level entries.
 - **Warning**: Uncheck to hide any warnings.
 - **Error**: Uncheck to hide any critical errors.
- **Limit visible events to**: Enter the maximum number of events to show in the Messages Pane. Default is 50.

Moving and merging panes

The PlanetPress Connect interface for both the Designer and DataMapper module is highly configurable. Each panel in the application can be moved, with the exception of the "The Data Viewer" on page 200 and "Workspace" on page 768 which area always in a static location. All panels can be minimized or maximized.

To move a panel:

- Click and hold the left mouse button on the panel title (tab) to move and keep the button pressed.
- Start moving the mouse to the new location. A grey outline shows where the tab will show up:
 - A small grey outline next to a current panel tab indicates that both tabs will be at the same location and only the active tab will display its content.
 - A larger grey outline at one of the edges of the Workspace or Data Viewer indicates that the separate will be separate and always visible.
- When the grey outline displays the location where the panel should be, release the mouse button.

To minimize a panel:

- Click the **Minimize panel** button at the top-right corner of the panel.

A minimized panel displays only as its icon wherever it was docked, generally on the left or right side, or the bottom.

To restore a minimized or maximized panel:

- Click the **Restore** button next to the panel's display icon.

The restored panel will return to its original docked location.

To temporarily display a minimized panel:

- Click the panel's display icon.

When another panel, menu or toolbar is clicked, the panel will be minimized again.

To maximize a panel:

- Click the **Maximize** button at the top-right corner of the panel.

A maximized panel takes the full available size for the panels. All other panels are minimized.

Outline pane

The Outline pane displays the current structure of the template, including all HTML tags present in each page.

- The display is in a treeview, the root being the *Page* node.
- At the top of the pane, a Text Filter box appears. Enter text in this box to only show elements which correspond to this inclusive filter. This can be class names, IDs, or element types (div, table, etc).
- Under the **Page** node, all top-level page elements are displayed. Each element under them is accessible by expanding (with the [+]) elements with children.
- Clicking on any element will select it in the [Workspace](#), whether it displays the Source, Design or Preview tab.
- Dragging an element inside the Outline pane re-orders it in the actual HTML. Elements are executed top-to-bottom with lower elements appearing on top of previous elements (unless a CSS Z-Index is used).
- Right-clicking an element displays a contextual menu offering the following option:
 - **Delete Element:** Click to delete the element from the outline view. This also removes it in the template itself for the current section.

Resources pane

The Resources pane displays the resources that affect the template and its output.

Tip

Images, fonts, stylesheets and snippets can be dragged or copied and pasted into the Resources pane to add them to your template.

Media

Media resources define paper handling configurations for Print output (see "Generating Print output" on page 956 and "Print Options" on page 851) including page size and paper type. See "Media" on page 353 for more information.

Contextual menu

- **New Media:** Click to create a new Media and open its properties ("Media Properties" on page 688).
- **Delete:** Click to delete the resource. This is the same as pressing the Delete key while the resource is selected.
- **Rename:** Click to open the resource's Rename. This is the same as pressing the F2 key while the resource is selected.
- **Properties:** Click to open the media properties.

Master Pages

Master Pages are layers of content that can be used by multiple Print Contexts to provide a reusable static background of content. Only one Master Page can be selected for each page position in the context. See "Master Pages" on page 350 for more information.

Contextual menu

- **New Master Page:** Click to create a new Master Page and open its properties.
- **Rename:** Click to open the resource's Rename. This is the same as pressing the F2 key while the resource is selected.
- **Delete:** Click to delete the resource. This is the same as pressing the Delete key while the resource is selected.
- **Properties:** Click to open the Master Page properties; see "Master Pages" on page 350 for more information.

Contexts

Contexts hold the actual content of the template that is used to generate output. See "Contexts" on page 320 for more information.

Contextual menu (Context folder or individual contexts)

- **New Print Context:** Click to create a new Print Context with a single section.
- **New Web Page Context:** Click to create a new Web Page Context with a single section.
- **New HTML Email Context:** Click to create a new HTML Email context with a single section.
- **Properties...** (Print and Email Contexts): Click to open the Context's properties. See "Contexts" on page 320 for more information.

Sections

Sections hold part of the contents within a specific context. See "Sections" on page 321 for more information.

Contextual menu

- **Set as Default** (Email and Web contexts only): Click to set the default section that is output if none is selected in the output generation.
- **New Section:** Click to add a new section within the context.
- **Rename:** Click to open the resource's Rename. This is the same as pressing the F2 key while the resource is selected.
- **Delete:** Click to delete the resource. This is the same as pressing the Delete key while the resource is selected.
- **Properties...:** Click to open the appropriate section properties: Email, Print or Web. See "Section properties dialogs" on page 719.
- **Includes...:** Click to open the "Includes dialog" on page 682.
- **Finishing...** (Print Sections only): Click to open the Finishing tab in the "Print section properties" on page 720
- **Sheet Configuration...** (Print Sections Only): Click to open the Sheet Configuration dialog; see "Master Pages" on page 350 and "Media" on page 353.

Images

Images are graphical elements that can be added to the page for display, either statically or dynamically. See "Images" on page 537 for more information.

Contextual menu

- **New Folder:** Click to create a new folder to organize resources more easily.
- **Rename:** Click to open the resource's Rename. This is the same as pressing the F2 key while the resource is selected.
- **Delete:** Click to delete the resource. This is the same as pressing the Delete key while the resource is selected.

Fonts

Font Resources included in a template are transported with it, so they can be accessed even if the template is moved to a different computer. Currently, fonts must be set through the CSS Stylesheet and do not appear in the fonts drop-down menu.

Currently supported font types: otf, woff, ttf, svg. Fonts must be set to *installable* to be useable in the output.

Please see the [Tips & Tricks post](#) for details on how to embed the fonts.

JavaScripts

JavaScripts are scripted programs that can run on Web output when added to the page header. See "Using JavaScript" on page 403 for more information.

Contextual menu

- **New Javascript:** Click to create a new JavaScript resource.
- **New Remote Javascript:** Click to add a Remote JavaScript resource. See "Using JavaScript" on page 403 for more information.
- **New Folder:** Click to create a new folder to organize resources more easily.
- **Rename:** Click to open the resource's Rename. This is the same as pressing the F2 key while the resource is selected.

- **Delete:** Click to delete the resource. This is the same as pressing the Delete key while the resource is selected.

Stylesheets

Stylesheets control how contents appears on the page. It defines spacing, color, size and other properties of elements on the page. See "Styling templates with CSS files" on page 553 for more information.

Contextual menu

- **New Stylesheet:** Click to create a new Stylesheet resource. Adding a new stylesheet will automatically include it in the currently active section.
- **New Remote Stylesheet:** Click to add a Remote Stylesheet resource. See "Styling templates with CSS files" on page 553 for more information.
- **New Folder:** Click to create a new folder to organize resources more easily.
- **Rename:** Click to open the resource's Rename. This is the same as pressing the F2 key while the resource is selected.
- **Delete:** Click to delete the resource. This is the same as pressing the Delete key while the resource is selected.

Snippets

Snippets are pieces of HTML or JSON code that can be inserted within sections and master pages, dynamically or statically. See "Snippets" on page 548 for more information.

Contextual menu

- **New HTML Snippet:** Click to create a new HTML Snippet resource.
- **New JSON Snippet:** Click to create a new JSON Snippet resource.
- **New Folder:** Click to create a new folder to organize resources more easily.
- **Rename:** Click to open the resource's Rename. This is the same as pressing the F2 key while the resource is selected.
- **Delete:** Click to delete the resource. This is the same as pressing the Delete key while the resource is selected.

Scripts pane

The Scripts pane contains all of the scripts that are used to replace data in a template, or to modify its look; see "Personalizing Content" on page 592.

This pane allows to add, edit and manage scripts (see "Managing scripts" on page 629). Scripts can be exported and imported via the buttons or through drag & drop between the Scripts pane and any location on the computer.

Note

Scripts included on the Scripts pane are completely distinct from the JavaScript resources found in the "Resources pane" on page 761 (see "Using JavaScript" on page 403).

Think of scripts as server-side in the sense that they are executed through the Connect modules (Server and Content Creation especially). Scripts have access to the whole PlanetPress Connect JavaScript API (see "Designer Script API" on page 874), such as the `record` object.

JavaScript resources, on the other hand, are only executed **after** the content creation is done, generally in a browser.

Note

The scripts in the Scripts pane are always executed top-to-bottom. They can be dragged up or down in the pane to change their order of execution. For example, content loading scripts (snippets with variable data, for instance) must come before scripts that replace data within that loaded contents.

Buttons

- **Import...:** Click to open a standard Open dialog to import a script. The script must have the .OL-script extension.
- **Export...:** Click to open a standard Save As dialog to save the currently selected scripts to disk. These scripts can be re-used in other templates. If more than one script is selected, they are all saved to a single file. If some scripts are inside folders, this folder structure is kept and will be restored when the scripts are imported.
- **New:** Displays a drop-down that shows the following options:
 - **Script:** Adds a new empty basic script.
 - **Text Script:** The default script that is created when adding variable data to a template. See "Variable Data" on page 604.

- **Dynamic Image script:** Provided that its selector refers to an image, this script dynamically changes the image for each record. See "Dynamic Images" on page 617.
- **Email scripts:** Email scripts define the sender, recipients, subject etc. of the email that is sent, and the PDF password. See "Email header settings" on page 373.
- **Control script:** A Control script affects the output of a template per record as a whole, instead of parts of the content. See "Control Scripts" on page 645 and "Control Script API" on page 930.
- **Conditional Content Script:** This script can conditionally show or hide any element in the template. See "Showing content conditionally" on page 613 and "Conditional script dialog" on page 718.
- **Folder:** Adds a folder in which scripts can be placed for easier management. See "Script folders" on page 629.
- **Collapse All:** Collapses all the folders, hiding the scripts inside of them.

Scripts Pane column

- **Name:** The name added to better identify the script.
- **Selector:** Displays the initial text or selector that the script applies to.

Note

Fields from the Data Model pane can be dragged directly into the Scripts pane to create a Text Script. Additionally, Text scripts can be dragged into any section to add the script's placeholder at the insert location. See "Variable Data" on page 604.

Contextual menu options

- **Duplicate:** Click to create an exact copy of the script.
- **Delete:** Click to delete the selected script. This does not delete any element or text in the template itself.
- **Rename:** Click to open a dialog to rename the script. This is the same as changing the **Name** field in the Edit Script window, which can be opened by double-clicking the script.

- **Enable/Disable:** Click to trigger the script to be enabled or disabled. Disabled scripts are greyed out and italic and will not be executed. See "Enable/disable scripts" on page 631
- **Import:** load a script from a Scripts file (*.OL-script).
- **Export:** save the script to a Scripts file (*.OL-script).
- **Properties** (Script folders only): edit the **name** and **execution scope** of the folder. See "Execution scope" on page 630.

Styles pane

The Styles pane shows which CSS style rules apply to the currently selected element. A link next to a style rule will open the file where that particular style is defined. This can be either a CSS file or the source file of a section if local formatting was used (see "Styling and formatting" on page 551).

A crossed-out style rule signals that it was overruled by another style rule. This happens when:

- A more specific, and therefore more important rule, is encountered for the same element. See "Using a more specific CSS rule" on page 560 to learn more about the specificity of style rules.
- A rule with the same importance is read after the first rule. Not only is the order of the rules in a CSS file important, but also the order in which the style sheets are read. The style sheets that are included with a section are read in the specified order; see "Applying a style sheet to a section" on page 323.

Workspace

The Workspace pane is where everything comes together. It is the contents of the page, the WYSIWYG editor that shows what the output will look like.

The Workspace contains three or, when a Web section is open, four tabs. To switch between the tabs, click on the tab at the bottom, or select **View > Design View, Preview View** or **Source View** on the menu.

For an overview of keyboard shortcuts, see "Keyboard shortcuts" on page 738.

Design tab

The Design tab shows the template including all styles, text and images as well as the placeholders used for variable data. In this tab, the template's scripts are not executed and only placeholders are shown.

The top of the Design tab contains an area with the following options and buttons:

- **Breadcrumbs:** Displays the element type where the cursor is located and any of its parent elements. Elements with classes or IDs show these details next to them, for instance `div #contents > ol.salesitems > li`. Click on an element in the Breadcrumbs to select it.
When an element is selected in the breadcrumbs and the Backspace or Delete key is pressed, that element is deleted.
If the deleted element was targeted by a script, you will be asked if you want to delete the script as well.
- **Context Selector:** Displays the current context. The drop-down lists available contexts. Clicking on a context switches to that context.
- **Section Selector:** Displays the currently active section. Clicking on another section switches to that section.
- **Media Selector** (Master Page editor only): Displays a list of Media resources. Clicking on a media will display its Virtual Stationery background while in Preview mode.
- **Zoom Level:** Displays the current zoom level and drops-down to change the level.
- **Buttons**
 - **Zoom in:** Zooms in by 25%
 - **Zoom out:** Zooms out by 25%
 - **Actual Size:** Zooms to 100%.
 - **Fit Width:** Adjusts zoom to fit the exact width of the template to the available workspace.
 - **Refresh:** Reloads the view, including static external images and remote stylesheets, and re-runs the scripts (the latter in Preview Mode only).
 - **Responsive Design View:** Use the drop-down to select a specific screen width, to test the design for different devices. Not available in Print contexts.

When a template is open, the workspace also shows a ruler and guides (see "Guides" on page 568).

Preview tab

The preview tab shows the template as it will output with the current record (see "Loading data" on page 594), with the personalized content (see "Personalizing Content" on page 592). Although it is possible to edit the template in Preview mode to a certain extent, it is recommended to do all editing in the Design mode.

Source tab

The source tab displays the HTML source for the template, including HTML Headers, CSS and HTML code. The source is displayed in a color-coded text editor, to quickly visualize the code. In this tab changes and adjustments can be made to the code. For shortcuts that can be used in this editor, see "Text editors: Source tab, JavaScript, CSS, Script Editor" on page 742 and "Emmet" on page 362.

To the left of the Source tab, a bar helps visually identify the start and stop of an element. For example when clicking on the opening `<table>` element, this bar marks the whole `<table>` and all its contents, until the ending `</table>` tag.

Pretty print options

In the Source view the HTML source of the template is "pretty printed" (that is: formatted, adding new lines and indentation) to make it more readable.

When this is undesirable, the Source view formatting can be turned off for (parts of) a section by adding `<!-- format:off -->` in Source view, at the beginning of the text or in between two HTML elements. From that point on, pretty printing will be disabled for that section. Use `<!-- format:smart -->` or `<!-- format:auto -->` to turn the formatting back on.

These are all `format` options:

- `<!-- format:off -->` turns the formatting off.
- `<!-- format:smart -->` or `<!-- format:auto -->` turns the formatting back on.
- `<!-- format:collapse -->` forces the following HTML elements to be collapsed.
- `<!-- format:expand -->` forces the following HTML elements to be expanded.

Example

This example shows how to turn pretty printing off for one specific HTML element: a Div element.

```
<p>In a "pretty printed" paragraph  
    <br>text is indented automatically on the Source tab.  
</p>  
<!-- format:off --><div anchor="page_media_0" style="font-family:  
Lucida Console,monospace; line-height: 1.2; white-space: pre-wrap;  
position: absolute; overflow: hidden; -moz-box-sizing: border-box;  
width: 675px; height: 68px; top: 209.967px; left: 65.4px;" offset-  
x="103.19999694824219" offset-y="247.76666259765625">@Header1@  
</div><!-- format:auto-->
```

Live tab

The Live tab shows the result of the template as rendered by the Gecko rendering engine. It is a good indication of how an HTML template would display in a visitor's browser, especially if they are using FireFox (which uses the Gecko engine).

Toolbars

This topic lists the buttons that are available in the top toolbar of the Designer module. For a description of the buttons at the top of the Workspace, see "Workspace" on page 768.

- **File Manipulation**

- **New:** Displays the New Wizard where a new data mapping configuration or a new template can be created.
- **Open:** Displays the Open dialog to open an existing template.
- **Save:** Saves the current template. If the template has never been saved, the Save As... dialog is displayed.
- **Print:** Opens the Print Output dialog.
- **Proof Print:** Opens the "Print Options" on page 851 dialog as a "Proof Print" which limits the number of records output. The options themselves are identical to the regular Print Output dialog.

- **Output**

- **Send Email:** Opens the Send Email dialog; see "Send (Test) Email" on page 723.
- **Send Test Email:** Opens the Send Test Email dialog; see "Send (Test) Email" on page 723.
- **Preview HTML:** Opens the current template's Preview in the system default browser. Useful for testing scripts and HTML output.

- **Send COTG Test:** Click to open the Send COTG Test dialog, to send the current Web Context to the Capture OnTheGo Application. See this how-to: [Testing a COTG template](#).
 - **Get Job Data File on submit:** Click to enable/disable. When enabled, the Job Data File will be returned to Connect Designer directly after a COTG Form has been submitted (see also: "Using COTG data in a template" on page 410).
- **Forms**
 - **Insert Form:** Inserts a <form> element.
 - **Insert Fieldset:** Insert a <fieldset> element.
 - **Insert Text Field:** Inserts a <input type="text" /> element. A drop-down is available to insert other fields, such as a URL, Password etc.
 - **Insert Text Area Field:** Inserts a <textarea> element.
 - **Insert Label:** Inserts a <label> element.
 - **Insert Checkbox:** Inserts a <input type="checkbox"> element.
 - **Insert Radio Button:** Inserts a <input type="radio"> element.
 - **Insert Select Field:** Inserts a <select> element and add multiple possible options to it.
 - **Insert Button:** Inserts a <button type="submit"> element at the current cursor location.

For information about Forms and Form elements, see "Forms" on page 527 and "Form Elements" on page 532.
 - **Pagination (Print Context only)**
 - **Page Number:** Inserts a placeholder for the current page number
 - **Page Count:** Inserts a placeholder for the total number of pages in the current section.
 - **Guides**
 - **Insert Horizontal Guide:** Click to insert a new horizontal guide; see "How to position elements" on page 567.
 - **Insert Vertical Guide:** Click to insert a new horizontal guide; see "How to position elements" on page 567.

- **Miscellaneous**

- **Insert Lorem Ipsum:** Inserts a paragraph of generic lorem ipsum text, useful for placeholder or template design.
- **Show Edges:** Shows a colored border around elements on the page and the type of element that is highlighted.

- **Form Wizard**

- **Form Wizard:** Click to open the Form Wizard to add a form to a Web Context. See "Forms" on page 527 and "Form Elements" on page 532.
- **Validation Settings:** Click to open the Validation settings dialog to change the validation settings on the currently selecting tools. See "Forms" on page 527.

- **Table Manipulation**

- **Insert Standard Table...:** Inserts a table with a specific number of columns and rows through the "Table" on page 542 Wizard.
- **Insert Dynamic Table...:** Inserts a dynamic table where the number of rows is determined by a Details table, through the "Dynamic table" on page 618 Wizard.

- **Select**

- **Select Table:** Selects the table where the cursor is located. If the cursor is within a table embedded within another, the innermost table is the one selected.
- **Select Row:** Selects the innermost row where the cursor is located.
- **Select Cell:** Selects the innermost cell where the cursor is located.

- **Delete**

- **Delete Table:** Deletes the innermost table where the cursor is located.
- **Delete Row:** Deletes the innermost row where the cursor is located.
- **Delete Column:** Deletes the innermost cell where the cursor is located.

- **Insert**

- **Insert Row Above:** Inserts a row above the current one. The row configuration, such as merged cells and cell styles, are duplicated, but contents is not.

- **Insert Row Below:** Inserts a row below the current one. The row configuration, such as merged cells and cell styles, are duplicated, but contents is not.
 - **Insert Column Before:** Inserts a column to the left of the current one. The column configuration, such as merged cells and cell styles, are duplicated, but contents is not.
 - **Insert Column After:** Inserts a column to the right of the current one. The column configuration, such as merged cells and cell styles, are duplicated, but contents is not.
- **Objects**
 - **Insert Image...:** Inserts an Image using a resource that is local to the template, at the current location of the pointer and opens its properties. See "Images" on page 537.
 - **Insert Image from Address...:** Inserts an Image using a URL instead of a resource, at the current location of the pointer and opens its properties. See "Images" on page 537.
 - **Insert Barcode:** Displays a list of available barcodes. Click on one to insert it on the page. See "Barcode" on page 470.
 - **Insert Pie Chart:** Click to insert a new Pie Chart object and open the Chart Script wizard.
 - **Insert Bar Chart:** Click to insert a new Bar Chart object and open the Chart Script wizard.
 - **Insert Line Chart:** Click to insert a new Line Chart object and open the Chart Script wizard.
 - **Hyperlinks**
 - **Insert Hyperlink...:** Creates a Hyperlink or mailto link on the currently selected text or element and opens its properties. See "Hyperlink and mailto link" on page 536.
 - **Remove Hyperlink:** Removes the currently selected hyperlink. The text or element that was the hyperlink is not removed.
 - **Boxes**
 - **Insert Positioned Box:** Inserts an absolute-positioned box on the page, which can be moved around freely.
 - **Insert Inline Box:** Inserts an inline box that is set to float to the left, at the position of the cursor.

- **Wrap in Box:** Takes the current selection and wraps it inside a new box.
 - **Float Left:** Floats the current element to the left using a *float:left* style.
 - **No Float:** Removes any *float* style applied to the currently selected element.
 - **Float Right:** Floats the current element to the right using a *float:right* style.
 - **Rotate Counter Clockwise:** Rotates the currently selected box 90° counter-clockwise.
 - **Rotate Clockwise:** Rotates the currently selected box 90° counter-clockwise.
- **Styles**
 - **Element Type:** Displays the element type of the selected element and drops down to show other element types in which it can be changed.
 - **Style:** Displays the style of the selected element and drops down to show other available styles which can be applied to it.
 - **Font Face:** Displays the font face of the selected text or element where the cursor is located and drops down to show other available font faces which can be applied to it.
Fonts added to the Fonts folder of the Resources pane are shown automatically in the Fonts drop-down.
 - **Font Size:** Displays the font size of the selected text or element where the cursor is located and drops down to show other available sizes which can be applied to it.
 - **Font Color:** When text is selected, click to apply the shown color to the selected text, or use the drop-down to change the color and apply it.
 - **Alignment**
 - **Align Left:** Aligns the currently selected element to the left.
 - **Align Center:** Aligns the currently selected element to the center.
 - **Align Right:** Aligns the currently selected element to the right.
 - **Justify:** Aligns the currently selected element to stretch text lines to fill all available width.
 - **Text Decoration**
 - **Bold:** Makes the currently selected text **bold**.
 - **Italic:** Makes the currently selected text *italic*.

- **Underline:** Makes the currently selected text underline.
- **Strikethrough:** Makes the currently selected text ~~strikethrough~~.
- **Indentation**
 - **Create Numbered List:** Makes the selected text element a numbered list (). If multiple paragraphs are selected, each becomes a list item (<li class="Bullet">).
 - **Create Bulleted List:** Makes the selected text element a bullet list (). If multiple paragraphs are selected, each becomes a list item (<li class="Bullet">).
 - **Indent:** Increases indentation of the selected text element. If the element is a paragraph, it is wrapped in a <blockquote> element. If it is a list item, it is moved to a child level, creating a new list if necessary.
 - **Outdent:** Decreases indentation of the selected text element. If the element is wrapped in a blockquote element, one blockquote is removed. If the element is a list item, it is removed from one surrounding list.
- **Position**
 - **Superscript:** Makes the currently selected text a ^{superscript}.
 - **Subscript:** Makes the currently selected text a _{subscript}.
- **Remove Formatting:** Remove any and all styles, text decorations and other formatting from the selected text. Indentation is not affected.
- **Welcome Screen:** Click to re-open the Welcome Screen.

Welcome Screen

The **Welcome Screen** appears when first starting up PlanetPress Connect. It offers some useful shortcuts to resources and to recent documents and data mapping configurations.

If you are new to PlanetPress Connect and you don't know where to start, see "Welcome to PlanetPress Connect 1.8" on page 14.

The Welcome Screen can be brought back in two ways:

- The **Welcome Screen** button in the "Toolbars" on page 771.
- From the Menus in **Help, Welcome Screen**.

Contents

- **Activation:** Click to open the **Objectif Lune Web Activation Manager**.
- **Release Notes:** Opens the current **Release Notes** for PlanetPress Connect.
- **Website:** Opens the PlanetPress Connect website.
- **Take A Tour:** Click to open the YouTube Playlist giving you a tour of the software.
- **Use the DataMapper to...:**
 - **Create a New Configuration:** Opens the [Creating a New Configuration](#) screen.
 - **Open an Existing Configuration:** Click to open the standard **Browse** dialog to open an existing data mapping configuration.
 - **Recent Configurations:** Lists recently used configurations. Click any configuration to open it in the DataMapper module.
- **Use the Designer to...:**
 - **Create a New Template:** Lets you choose a Context to create a new template without a Wizard.
 - **Browse Template Wizards:** Displays a list of available Template Wizards, producing premade templates with existing demo content; see "Creating a template" on page 304.
 - **Open an Existing Template:** Click to open the standard **Browse** dialog to open an existing template.
 - **Recent Templates:** Lists recently used templates. Click any template to open it in the Designer module.
- **Other Resources:**
 - **Documentation:** Opens this documentation.
 - **Courses (OL Learn):** Opens the [Objectif Lune e-Learning Center](#).
 - **User Forums:** Opens the [Questions & Answer](#) forums.

Print Options

The **Print Options** page is the first page of both the **Advanced Print Wizard** and the [Output Creation Settings](#) Preset .

This page is the most important of the Advanced Print Wizard.

The other pages that appear throughout the Wizard are determined by the selections made on this page.

The choices can be broken down as follows:

- **Printer** group:

- **Model:** Use the drop-down to select the printer language / output type that will be generated.
Connect output options cover a range of industry standard print output types. These include PCL, PDF and PostScript (including PPML, VIPP and VPS variants), with a range of quality settings available.

Note


By default, Connect displays only the PDF output option, but other print output types can be added to the Printer Model drop down list via the Settings button



For more information on how to do this, see "Adding print output models to the Print Wizard" on page 960.

- **Output Options** group:

- **Output Local** checkbox: Select to have the output created using the local Print Server, instead of the Connect Server (see "The Connect server" on page 95 for an explanation).
- **Output Type** choices:
 - **Prompt for file name:** Select to output to a local file on the hard drive. When this option is selected, no other configuration is necessary. A Save As dialog will appear to allow selection of the folder and filename.
 - **Directory** : Select to output to a local folder on the machine.
Selecting this will open the **Directory Options** sub-group, which has these options:
 - **Job Output Mask:** The name of the file that will output.
You could write the Job Output Mask directly into this edit box (for a list of available variables see "Print output variables" on page 961), or you

could create a Mask via the Options  button. This opens the custom dialog: Job Output Mask Dialog.

The Job Output Mask may contain (dynamic) folder names, for example: `${document.metadata['Country']}\${template}`. The evaluated value of the Job Output Mask is taken as a path **relative** to the folder specified by the Job Output Folder (the next option in this dialog). The Job Output Folder must exist, but folders specified in the Job Output Mask will get created if they don't exist.

- **Job Output Folder:** The path on the disk where the file is produced. Please note that the folder must exist, or output will fail when produced through the server.
- **LPR Queue:** Select to send the print job to an LPR queue. It is assumed that the print technology is supported by the system receiving the LPR job.
 - **Local Printer:** The IP or host name of the printer or machine where the LPD is installed and will receive
 - **Queue Name:** The queue name that will accept the job on the LPD. Default is generally "auto".
 - **Job Owner Name:** Optional entry for adding the name of the job owner.
 - **Job Name:** The name of the output file. You can use `${template}` as a variable for the name of the Designer Template used to generate the output.
- **Windows Printer:** Select to send the Print Job to a Printer Queue. The job is rendered as a PDF before being printed through the Windows driver.
 - **Windows Printer:** Use the drop-down to select the windows printer queue where the job will be sent.
 - **Job Owner Name:** Optional entry for adding the name of the job owner.
 - **Job Name:** The name of the output file. You can use `${template}` as a variable for the name of the Designer Template used to generate the output.
- **PDF Rendering Options (PDF output only):**
 - **Auto-rotate and center:** Check to automatically select the page orientation that best matches the content and paper.

- **Choose paper source by page size:** Check to use the PDF page size to determine the output tray rather than the page setup option. This option is useful for printing PDFs that contain multiple page sizes on printers that have different-sized output trays.
- **Scale:**
 - **None:** Select to not scale any page, whether it fits or not.
 - **Expand to printable area:** Select to expand any page to fit the page area. Pages larger than the paper size are not resized.
 - **Shrink to printable area:** Select to shrink any page to fit the page area. Pages smaller than the paper size are not resized.
- **Production Options:**
 - **Booklet Imposition** checkbox: Check to tell the printer to generate a booklet for the print output. Booklet options are set in the "Booklet Options" on page 857 page. This option is unselected by default unless selected in the Designer "Print section properties" on page 720.
 - **Cut and Stack Imposition** checkbox: Check to enable Cut & Stack Imposition, which is set in the "Imposition Options" on page 858 page.
 - **Add Inserter marks** checkbox: Check to enable inserter mark functionality, which is set in the "Inserter Options" on page 865 page.
 - **Override Finishing options** checkbox: Check to configure custom "Finishing Options" on page 841, such as binding.
 - **Print virtual stationery** checkbox: Check to enable virtual stationery in the output.
 - **Use grouping** checkbox: Check to configure grouping of output into jobs, job segments or document sets. See "Grouping Options" on page 847.
 - **Include meta data** checkbox: Check to add meta data to the output. This can be done at Job, Job Segment, Document, Document Set and Page level. See "Metadata options" on page 849.
 - **Separation:** Check to activate the "Separation options" on page 856 page of the wizard.
 - **Add additional content** checkbox: Check to activate the "Additional Content" on page 799 page of the wizard.

- **Records** group:
 - **Record Range:** Allows selection of a range of records or a custom selection. You can specify individual records separated by semi-colons (;) or ranges using dashes. For example: 2;4;6-10 would print pages 2, 4, 6, 7, 8, 9 and 10.
- **Copies** section:
 - **Copies:** Enter the number of copies to print, of each record.
 - **Collate:** When printing multiple copies you can check this checkbox to have the record copies printed together. For example in a three record job the records would print out as 1-1-2-2-3-3, rather than 1-2-3-1-2-3.
- **Pure Color Thresholds** group:

This section is valid for PCL only. It applies to elements within the record that are shades of gray, rather than black or white.

 - **Black Threshold Percentage:** The percentage of shading at which the element will appear as full black, rather than dark gray.
 - **White Threshold Percentage:** The percentage at which the element will appear as full white, rather than light gray.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 777 page) and add or remove printing options from the print run.
- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 777 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Printer Settings

The **Printer Settings** page defines options on the printer. It is available for PostScript (and the VIPP and VPS variants of PostScript) only.

- **Map media by** options: Select from following choices:
 - **Media Attribute** displays all Media details, except the Tray selection.
 - **Tray** displays just the Media name and Tray selections.
 - **Both** displays all Media details.
- Tray selection columns:
 - **Media**: Lists the Media name, as defined in the template.
 - **Tray**: Use the drop-down to select in which tray to send any page using the media.
 - **Position**: Enter a MediaPosition option on the printer to define the media to use.
 - **Weight**: Enter a weight for the paper.
 - **Type**: Use the drop-down to select which type of stock to use on the printer.
 - **Color**: Use the drop-down to select which color the paper should be on the printer.

Booklet Options

The **Booklet Options** page defines how to generate booklets in the output. It is used in conjunction with [Imposition](#) settings, which will appear after the Booklet entries have been made.

This page includes a handy illustration that displays how the final binding would look, based upon the current selections.

Options:

- **Configuration**: Use the drop-down to select the type of binding to use:
 - **Saddle Binding**: This binding places all the pages in a stack, binds the middle and folds the stack as one.

- **Perfect Binding:** This binding type is often used for books. Pages are folded in the middle and then set side by side. The pages are then bound along the folded "spine".
- **1 up Perfect Binding:** This binding does not contain any folding. The pages are lined up side by side and bound along one edge.
- **Booklet Binding Edge:** Use the drop-down to select the side on which to bind the booklet.

Optional **Cover Page** selections are available to Saddle Binding only.

- **Cover Page** checkbox: Check to enable cover pages to be created with the options below:
 - **Media** selections:
 - **Cover Media Size:** Use the drop-down to select the media size for the cover page, or use a Custom size and select **Width** and **Height** values.
 - **Front Cover** selections:
 - **Blank:** Select to add no data to the front cover.
 - **First page on outside and second page on inside:** Select to use the first 2 pages as the inside and outside of the front cover.
 - **Back Cover** selections:
 - **Blank:** Select to add no data to the back cover.
 - **Last two pages on inside and outside:** Select to use the final 2 pages as the inside and outside of the back cover.

Imposition Options

Imposition refers to the printing of multiple pages on a single sheet. This is also known as N-Up printing.

The options on the **Imposition Options** page allow for the setting of imposition repetition, order, margins and markings.

As Imposition selections are very specific and can be quite confusing, we have provided a handy dynamic diagram that displays a representation of the current Imposition selections in real time. Whenever selections are changed, this display changes to reflect the selections made.

You can even select which specific pages are to be represented (up to page 1,000), whilst page numbers appear on the pages in the diagram and change dynamically, like everything else.

The Imposition selections that can be made are as follows:

- **Sheet Size** group:

- **Final Media Size:** Use the drop-down to select the size of the media where the output is printed. The size of the media should be equivalent to the initial Section size multiplied by the number of repetitions, added with the margins and spaces between the repetitions.

If *Custom* media size is selected, enter the custom **Width** and **Height** values.

Note

The Sheet Size cannot be altered if a Cover Page was selected in the "Booklet Options" on page 857 Page.

- **Orientation:** Select orientation (aspect ratio) of media (Landscape or Portrait), or allow Connect to automatically determine the proper aspect ratio (Auto-Rotate).
- **Position:** Select from following options:

Note

If "Booklet Options" on page 857 were selected, then the **Position** settings are pre-set and cannot be altered here.

- **Auto-positioned:** This option creates unscaled imposition-ed pages.
- **Scale to fit:** Scales the imposition-ed pages so they fit on the N-Up stock. The scaled pages are then auto-positioned as usual.
- **Offset:** Allows for the selection of an offset position. The imposition-ed pages will be laid out so that the top left corner of the top left imposition-ed page is located at the selected offset.
If Offset is chosen, then the **Left Offset** and **Top Offset** selection boxes

become active.

Note

The offset measures from the top left of the physical N-Up sheet to the top left imposition-ed page. If Auto-Rotate is selected (causing the N-Up stock to be rotated to fit the imposition-ed pages) then the measurement becomes the top left position of the rotated stock. i.e. The top left corner does not rotate with the stock.

- **Rotate final output Sheet 180 degrees (upside down):** Select to flip the output upside down.
- **Repetition** group:
Allows selection of how many Sections are to be placed, both Horizontally and Vertically. This is the total number of items, not the number of additional items being placed.

Note

If *Booklet Binding* were selected, some of these settings will be determined by the options made within the "Booklet Options" on page 857 Page and they cannot be altered here.

- **Gap** group:
Allows selection of the amount of blank space (either **Horizontal** and/or **Vertical**) to add between each page.

Note

If *Booklet Binding* were selected, some of these settings will be determined by the options made within the "Booklet Options" on page 857 Page and they cannot be altered here.

- **Order group:**

Note

If *Booklet Binding* were selected, some of these settings will be determined by the options made within the "Booklet Options" on page 857 Page and they cannot be altered here.

- **Page Order:** Select in which direction to go when adding sections to the output:
 - **Left to right, then top to bottom**
 - **Right to left, then top to bottom**
 - **Top to bottom, then left to right**
 - **Top to bottom, then right to left**
- **Stack Depth:** Stack Depth defines the number of sheets that will be printed before drill sorting resets.

Stack Depth works in conjunction with the Repetition, Gap and Page Order selections in order to determine the printing order

It is best illustrated by way of example.

The following example has a job with 100 single page documents and Repetition settings of 5 Horizontal and 2 Vertical, plus a Stack Depth of 5. Each table represents a single sheet, and the red line between the tables represents the Stack Depth:

1	6	11	16	21
26	31	36	41	46
2	7	12	17	22
27	32	37	42	47
3	8	13	18	23
28	33	38	43	48
4	9	14	19	24
29	34	39	44	49
5	10	15	20	25
30	35	40	45	50
51	56	61	66	71
76	81	86	91	96
52	57	62	67	72
77	82	87	92	97

Each set of 5 sheets is a discrete drill sorted nUp block that can be guillotined and stacked. The first 5 sheets encompass records 1-50, whilst the next 5 sheets encompass records 51-100.

The following example is the same job but with a Horizontal Repetition setting of 6:

1	6	11	16	21	26
31	36	41	46	51	56
2	7	12	17	22	27
32	37	42	47	52	57
3	8	13	18	23	28
33	38	43	48	53	58
4	9	14	19	24	29
34	39	44	49	54	59
5	10	15	20	25	30
35	40	45	50	55	60
61	66	71	76	81	86
91	96				
62	67	72	77	82	87
92	97				

As you can see, the outputs are *very* different.

Note

If the Stack Depth is greater than the number of sheets in the entire job then you will get a single drill sorted block that can be guillotined and stacked to give the complete job in record order.

- **Reverse Pages:** Select this option to reverse the order of pages. This would print the final record on the first page and the first record on the last page.
- **Force simplex:** Select this option to make the output Simplex, rather than the Imposition default of Duplex.

Note

If *Booklet Binding* were selected, some of these settings will be determined by the options made within the "Booklet Options" on page 857 Page and they cannot be altered here.

- **CropMarks** group:
 - **Type:** Use the drop-down to select the type of Crop Marks to add to the page.
 - **Page side:** What side(s) of the page to put the Crop Marks.
 - **Width:** Select the width of the crop mark lines.
 - **Offset:** How much separation (if any) to leave between the vertical and horizontal corner markings.
 - **Length:** Select the Length of the crop mark lines.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.

- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.
- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.



Insertor Options

The **Insertor Options** page allows the selection of a High Capacity Feeder (HCF) model. These machines are also commonly referred to as Inserters or Folder-Inserters.

The options available on this page are dependent upon the model selected.

The options selected on this page influence the position of the markings set on the next page: **"Mark Position Options" on page 867.**

- **Model:** Use the drop-down to select from any previously loaded Inserter model, or use the Browse button to select a HCF file to load a new Inserter model.
An image representing the chosen folder-inserter is displayed under the list, along with the HCF file details.
- **Options Group:**
The options available here are all Inserter dependent, and thus will change based upon the Inserter model selection.
To see how the selected Inserter markings would look on the printed page, click the Next button to move to the "Mark Position Options" on page 867 page, which has a preview of the page. You can move back and forward between these two pages until you are entirely satisfied with the selections made.
 - **Mark Configuration:** Use the drop-down to select the type of markings to add. This selection basically equates to the amount of area the markings will take up on the printed page.
 - **Fold Type:** Use the drop-down to select the type of fold to apply to the paper. This will impact upon where on the page the markings will be placed.
 - **Collation level:** Select whether the markings will be made at Document level, or Document Set level.
 - **Print marks on back:** Check to place the Inserter Marks on the rear of the page.

- **Selective Inserts:** If selective inserts are supported by the chosen Mark Configuration you can select what markings to include and whether those markings are to included based upon some conditional setting. This can be done by highlighting the Mark Name entry and either pressing the  **Edit** button, or using the right mouse click context menu, and selecting  **Edit**.
For information on how to edit the Selective Inserts settings, see Selective Insert Dialog
- **Clear Background Area Tab:**
Check the **Clear Background Area** checkbox to add a white background to the OMR, preventing background colors or elements interfering with the OMR Markings when they are read by the Inserter.
 - **Margins:**
 - **Same for all sides:** Check so that the Left margin selection is used to set all sides identically.
 - **Left, top, right, bottom:** Enter measurements for the margins on each side of the OMR Marks.
- **Custom OMR mark sizing Tab:**
If supported by the currently chosen Mark Configuration you can select a Custom OMR size by checking the **Custom OMR mark sizing** checkbox.
Select from any of the following, or leave the entries blank to use default values:
 - **Line length:** Enter a value between 10.16mm and 20mm.
 - **Line thickness:** Enter a value between 0.254mm and 0.63mm.
 - **Gap distance:** Enter a millimeter value 2.91mm and 4.2mm.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.

- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Mark Position Options







The **Mark Position Options** page displays a Preview of the output and the possible locations to place the inserter marks. The initial settings are determined by the selections made within the "Inserter Options" on page 865 page.

You can move back and forward between these two pages to perfect the settings, or you could move the inserter mark box to the desired location on the preview.

Preview box:

- The *pink area* displays the areas of the page where inserter marks can be positioned.
- The *small checkered box* displays the current location of the inserter marks. This box is selectable and can be dragged to the desired location within the printable (pink) areas. If the box is placed outside the printable areas the page will display an error and prevent attempts at leaving the page.

Below the Preview box are buttons which allow control of the Preview box. The selections that can be made are:

-  **First Page**: Click to jump to the first page.
-  **Previous Page**: Click to move to the previous page.
-  **Next Page**: Click to move to the next page.
-  **Last Page**: Click to jump to the last page.
- **Show Page**: Use the up and down arrows or type a page number to display a specific page within the document.
-  **Zoom in/out**: Click to zoom in or out by 25%
-  **Zoom Level**: Use the drop-down to select a predefined level or enter a zooming percentage.






Finishing Options

Use the **Finishing Options** page to force the use of specific printer finishing options, rather than using any finishing options that might have been set in the Template's Media and Section options.

These settings are only applied when producing print output. They do not modify the original finishing options in either the Section or the Template.

Finishing settings can be set on various levels of Job Creation, such as Document Sets and Job Segments. This allows you, for example, to staple Document Sets whilst punching holes in Job Segments.

You can also have multiple finishing settings at the same level.

- **Finishing Options Table:** This table, on the left of the Wizard page, allows you to add or remove sections to apply the Finishing options to. It contains the **Section** for which the settings are to be applied (*Section, Document, Document Set* or *Job Segment*), plus the **Finishing** options that are to be applied, and any **Rule** determining the conditions under which the Finishing settings are to be applied. To set the details, use the following options:
 -  **Import Finishing:** Imports the existing settings from a Connect Template. The Template could be the current template (***Import from currently template***), or some other Template (***Import from template ...***).
 -  **Add:** Adds a new section to apply Finishing options to. The default is to add a new *Section*, but these can be changed to *Document, Document Set* or *Job Segment* types, thereafter. You can have multiple selections of each type, all with different criteria determining when the Finishing options are to be applied.
 -  **Delete:** Removes the selected Section(s) from the table.
 -  **Move Up / Move Down:** Moves the selected Section(s) up or down within the table.
 -  **Edit ... :** Brings up the Rule Editor Dialog, which can be used to construct the rule (s) that determines whether this conditional Finishing option is to be applied or not.
- **Binding group:**
 - **Style:** What type of Binding to request on the printer. This includes *Stapled, Glued, Stitched, Ring, Comb, Coil*, amongst other options..

- **Side:** Sets the side of the paper that the Binding is to occur. This includes both the top and the bottom of the paper,
- **Location:** Sets where the binding is to occur, if applicable.
The selections available here will be dependent upon the selection made in the Binding **Style**. Only Stapled and Stitched bindings have a **Location** option available to them.
- **Angle:** Set Stapling or Stitching binding either horizontally, vertically, or at an angle (as supported by printer).
- **Item count:** Select the amount of Staples or Stitches to use. The choice is between the default amount or selecting a specific number using the Count option.

Tip

The options available to you in reality at print time will be printer dependent, so you will need to know the capabilities of your printer, or leave the value set to Default.

- **Area:** The area where the binding can be applied.
- **Hole making group:**
Hole making options are available only to *Ring*, *Comb* (wire and plastic) and *Coil* Binding **Styles**. The selections will need to be made at run-time based upon the types of binding options available that the printer supports.
 - **Number of holes:** The number of holes to punch for the selected Binding option.
 - **Pattern Catalog ID:** The Catalog ID of the selected Binding option.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way

through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.

- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Grouping Options

The **Grouping options** page separates the job output into multiple blocks that can then be physically separated using split sheets in the printer.

- **Grouping Tabs:** Jobs can be grouped at three different levels. The three groups each have their own tab, and are as follows:
 - **Job** Grouping Fields
 - **Job Segment** Grouping Fields
 - **Document Set** Grouping Fields

The Fields available to be used for Grouping are contained within the **Available Fields** box.

Any Fields that you want to use for Grouping need to be added to the **Selected Fields** box via the arrows found between the two boxes.

Simply select the Field(s) you want to move and then click the appropriate arrow.



Any fields that you decide don't need to be used in Grouping can be returned to the Available Fields box in the same fashion.

Once a field is added to the **Available Fields** box, the *Sorting Option* can be selected by clicking in the "**Sorting Option**" column, and selecting the appropriate option. The selection can be made between Ascending or Descending order, or Unsorted.

- **Page Break Grouping:** Check this check box to enable page break grouping, which separates Documents into different groups, based on the number of pages they contain. For example, enabling the *Document Set Grouping* Level and creating a page range from 1-5 and 6 to Largest, will create two Document Set groups. The first will contain all Documents of 1 to 5 pages length, the second will contain any document of 6 or more pages.

Note

Page Break Grouping works only on Document page counts.

- **Grouping Level:** Use the drop-down to select which grouping level to use, between **Job**, **Job Segment** or **Document Set**. Only one grouping level can be selected.
- **Grouping list:** Add  (or remove ) entries to this list to create new groups based upon the number of pages in the level selected above. All groups must be contiguous from 1-to-Largest and they must not contain any gaps.
 - **Range Name:** Enter a name identifying the range. It must be unique, but otherwise bears no impact on the range feature.
 - **From:** Enter the starting page number of the range. The first range must start with 1, all other ranges must be contiguous (the "From" range must be one higher than the previous "To" value).
 - **To:** Enter the last page number for the range. The last range must end with a selection of "Largest".
- **Generate page break ranges in reverse order:** Reverses the order of the groups created.

By default, grouping is done from smallest to largest. Checking this option instead creates groups from largest to smallest.
- **Generate page break range groups after normal grouping:** Check this option to first group using the levels above, following which page break grouping are applied. This creates two different levels of grouping, applied in order.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.

- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Metadata options

The **Metadata Options** page defines metadata tags that will be added to the output file when producing PDF and AFP output in the "Output Creation Settings" on page 850.

Metadata tags are ignored in all other output types, except when they are associated with "Additional Content" on page 799 in the Print Wizard.

The tags can be added to any of these levels, as indicated by the tabs on top: **Job**, **Job Segment**, **Document**, **Document Set**, and **Page** Tags.

In each of these levels, a list of tags is available:

- **Always create meta data for this level even when fields are selected:** Select to create a blank meta data entry if no fields are selected. Done to ensure that a metadata store is always available, if required.
- **Tag Name:** Name of the metadata tag added to this level. Once a tag has been added, its name can be edited by double-clicking on the Tag Name.
- **Source Type:** Displays the type of field being used - either Text or Data Field.
- **Source:** For Data Fields only. The Field name from the data mapping configuration whose value will be used for this tag.
- **+ Add Field:** Click to add a new tag to the current level. The Field Selection dialog appears. Select either **Add field meta data** or **Add text meta data**.
When adding field meta data select a field name from the Field List and click OK to add it as a tag of the same name.
- **- Delete Field:** Click to delete the currently selected tag.
- **↑ Move Up:** Click to move the currently selected tag one position up.
- **↓ Move Down:** Click to move the currently selected tag one position down.

Separation options

The **Separation Options** page defines how to separate the jobs using subsets, slip sheets, or jogging.

- **Sheet Count Splitting** group.

This group allows for the splitting of output based upon a pre-determined number of pages

- **Split:** Use the drop-down to select how to split.
 - **None:** Select to ignore sheet count splitting entirely.
 - **At exactly:** Select to create a split at a specific sheet number.
- **Every:** Enter the number of sheets at which to split the output.

- **Separation Settings** group.

This setting is only available if no Sheet Count Split were specified.

- **Separation:** Use the drop-down to select when a job separation occurs, which is either **None** (no separation) or at the **Job, Job Segment, Document** or **Document Set** level.

- **Slip Sheets** group

- **Add slip sheet:** Use the drop-down to select whether to add a slip sheet before or after a specific separation, or whether to use none.
- **Every:** Use the drop-down to select at which separation to add a slip sheet, at the **Job, Job Segment, Document** or **Document Set** level.
- **Media Size:** Use the drop-down to select the media size of the slip sheet.
If a custom Media Size was chosen:
 - **Width:** enter slip sheet page width.
 - **Height:** enter slip sheet page height.

- **Jog** group

- **Jog after every:** Use the drop-down to select when to jog the printer, which is either **None** (no forced jogging) or at the **Job, Job Segment, Document** or **Document Set** level.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.
- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Additional Content

The **Additional Content** page allows you to add content at the time of printing. There are four different types of Additional Content that can be added at print time: **Text**; **Images**; **Barcodes** and **OMR Marks**.

They can be used to add either static or variable content.

The Additional Content option is particularly useful when you might need to drive custom processes on production machines using either Barcodes or OMR Marks, or if you need to add some last minute additions to the print job via text and/or images.

Additional Text





Text is added to the output at specific positions. This dialog displays all the text settings:

- **Left**: Displays the distance between the left margin of the page and the text.
- **Bottom**: Displays the distance between the bottom margin of the page and the text .
- **Orientation**: Displays the orientation of the text.

- **Text:** Displays the actual text.





Note

The entered text might have been entered over multiple lines, so not all of the text will be displayed here. You might consider this a text entry preview of the text, rather than the complete text entry.

- **Condition:** Displays the condition which is used to determine if text element is to be included or not.
-  **Add:** Click to open the [Additional Text Settings](#) dialog to add a new text entry.
-  **Delete:** Click to delete the currently selected entry.
-  **Edit:** Click to edit the currently selected entry using the [Additional Text Settings](#) dialog.
-  **Duplicate:** Click to create a copy of the entry.





Additional Images

Images are added at specific positions, with optional dimension constrains. This dialog displays all the configured additional image settings:

- **Left:** Displays the distance between the left margin of the page and the image.
- **Bottom:** Displays the distance between the bottom margin of the page and the image .
- **Orientation:** Displays the orientation of the picture.
- **Filename:** Displays the selected image filename.
- **Condition:** Displays the condition which is used to determine if the image is to be included or not.
-  **Add:** Click to open the "Image Settings" on page 803 dialog to add a new text entry.
-  **Delete:** Click to delete the currently selected entry.
-  **Edit:** Click to edit the currently selected entry using the "Image Settings" on page 803 dialog.
-  **Duplicate:** Click to create a copy of the entry.



Additional Barcodes



Barcodes are added at specific positions. This dialog displays all the configured additional Barcode settings:

- **Left:** Displays the distance between the left margin of the page and the Barcode .
- **Bottom:** Displays the distance between the bottom margin of the page and the Barcode .
- **Orientation:** Displays the orientation of the Barcode .
- **Type:** Displays the type of Barcode that's added.
- **Text:** Displays the data used for generating the barcode content.
- **Condition:** Displays the condition which is used to determine if the barcode is to be included or not.
-  **Add:** Click to add a Barcode. Select from the list of Barcode types that appears. The [Additional Barcode Settings](#) page lists all the available Barcodes, and links to their options..
-  **Delete:** Click to delete the currently selected Barcode entry.
-  **Edit:** Click to edit the currently selected Barcode entry using the [Additional Barcode Settings](#) dialog.
-  **Duplicate:** Click to create a copy of the barcode entry.

Additional OMR Marks

Optical Mark Recognition (OMR) marks are added at specific positions, with optional dimension constrains. This dialog displays all the configured additional image settings:

- **Left:** Displays the distance between the left margin of the page and the OMR mark.
- **Bottom:** Displays the distance between the bottom margin of the page and the OMR mark.
- **Orientation:** Displays the orientation of the OMR mark.
- **Condition:** Displays the condition which is used to determine if the OMR mark is to be included or not.
-  **Add:** Click to open the "OMR Mark Settings" on page 830 dialog to add a new text entry.
-  **Delete:** Click to delete the currently selected entry.

-  **Edit**: Click to edit the currently selected entry using the "OMR Mark Settings" on page 830 dialog.
-  **Duplicate**: Click to create a copy of the entry.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.
- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Additional Text Settings

The **Additional Text Settings** dialog displays the property of Text added in the "Additional Content" on page 799 page.

- **Position group**:
 - **Orientation**: Use the drop-down to select the orientation of the Text added to the page.
 - **Output once per sheet**: Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Text printed once per sheet rather than once per document page.

Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.


- **Left:** Enter the distance between the left margin of the page and the Text, in either metric (cm/mm), inch (in), pixel (px) or point (pt) values.
- **Bottom:** Enter the distance between the bottom margin of the page and the Text, in either metric (cm/mm), inch (in), pixel (px) or point (pt) values.
- **Font group:**
 - **Font Name:** Use the drop-down to select which font type to apply to the Text. The drop-down displays all the fonts installed on the system.
 - **Font Size:** Enter the font size in points (pt).
 - **Bold:** Check to make the Text **bold**.
 - **Italic:** Check to make the Text *italic*.
 - **Color:** Select what **color** the Text will be.
- **Text:** Enter the actual Text to appear on the page in the selected location. The Text can be spread over multiple lines, but no additional formatting can be added within this edit box. The entire Text will be printed use the formatting options selected in the **Font group**.
 -  **Add:** Click to display a list of variable data that can be added to the Text. This includes metadata fields added in the [Metadata Options](#), as well as some document information fields.
- **Condition:** Enter the condition which determines whether or not the Text will be added to the document at print time.
For details on how to create a conditional, see the Conditionals page.

Image Settings

The **Additional Image** dialog displays the properties of the image added in the "Additional Content" on page 799 page.

- **Position group:**
 - **Orientation:** Use the drop-down to select the orientation of the image.
 - **Layer:** Whether this image will appear behind the text (the text will print over the image) or in front of the text (the text behind will be blanked out by the image, as transparent images are not supported)
 - **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Image printed once per sheet rather than once per document page.

Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.

- **Left:** Enter the distance between the left margin of the page and the image, in either metric (cm/mm), inch (in), pixel (px) or point (pt) values.
- **Bottom:** Enter the distance between the bottom margin of the page and the image, in either metric (cm/mm), inch (in), pixel (px) or point (pt) values.
- **Filename:** Use the browse button to select an image.

Note

Transparent images are not supported.

- **Scaling group:**

Scaling the image expands the image but keeps the aspect ratio. The amount of scale and specific limitations can be applied used a combination of the following options:

 - **Max Width:** Enter the absolute maximum width the image can be scaled to, in either metric (cm/mm), inch (in), pixel (px) or point (pt) values.
 - **Max Height:** Enter the absolute maximum height the image can be scaled to, in either metric (cm/mm), inch (in), pixel (px) or point (pt) values.

- **Scale:** What scale to apply to the image. The maximum scale is 10.0 to 1. Decimal values are allowed for this field.
- **Condition:** Enter the condition which determines whether or not the image will be added to the document at print time.
For details on how to create a conditional, see Conditionals page.

Barcode Options

When adding Barcodes in the "Additional Content" on page 799 page you can select from a series of predetermined Barcode types.

Note

To create dynamic barcodes, "Metadata options" on page 849 must first have been set. Metadata fields are required to create the association between the dynamic data used in the print run and the barcode.

The options for each of these types is described on the following pages:

- Australia Post 4 State Settings
- "Codabar Settings" on the facing page
- "Code 128 Settings" on page 808
- "Code 39 Settings" on page 809
- "Data Matrix Settings" on page 812
- "EAN-128 Settings" on page 813
- "EAN-13 Settings" on page 815
- "EAN-8 Settings" on page 817
- "Interleaved 2 of 5 Settings" on page 819
- KIX Code (Dutch Post) Settings
- "PDF417 Settings" on page 821
- "QR Code Settings" on page 823
- Royal Mail Settings
- Royal Mail 2D Settings

- "UPC-A Settings" on page 826
- "UPC-E Settings" on page 828
- US Postal Service IMB Settings
- US Postal Service IMPB Settings

Codabar Settings

Codabar barcodes support the following data: 0-9 - \$: / . + plus the optional specification of start/stop characters.

Note

Note

To create dynamic barcodes, "Metadata options" on page 849 must first have been set. Metadata fields are required to create the association between the dynamic data used in the print run and the barcode.



Use the following options to configure the output Barcode settings:

- **Position** group:
 - **Orientation:** Use the drop-down to select the orientation of the Barcode added to the page.
 - **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Barcode printed once per sheet rather than once per document page.

Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.

- **Left:** Enter the distance between the left margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.

- **Bottom:** Enter the distance between the bottom margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
- **Codabar Properties** group:
 - **Height:** Enter the Barcode height in either metric (cm/mm), inch (in) or point (pt) values.
 - **Module Width:** Specifies the width of the narrow bars. Changing this value to higher value will generally make the Barcode bigger.
 - **Bar width ratio:** Set the Barcode bar width.
 - **Default start symbol:** Use the drop-down to select the optional Barcode start character, which defines the encoding mode.
 - **Default stop symbol:** Use the drop-down to select the Barcode stop character, which defines the encoding mode.
 - **Print human readable text:** Check to add a textual version of the Barcode data.
 - **Placement:** Use the drop-down to select whether to place the human readable text above or below the Barcode.
 - **Font name:** Use the drop-down to select the font with which to display the human readable text.
 - **Font size:** Enter a font size for the human readable text.
 - **Display start/stop symbols** check box: Adds the stop/start symbols to the Barcode text.
- **Text:** Enter the text used to generate the Barcode.
 -  **Add** button: Click to display a list of variable data that could be used for generating the Barcode. This includes metadata fields which must previously have been added in the [Metadata Options](#) (likely at the *Document Tags* level), as well as some information fields.
- **Condition:** Enter the condition which determines whether or not the Barcode will be added to the document at print time. For details on how to create a conditional, see the Conditionals page.
 -  **Add** button: Click to display a list of metadata fields, information fields to add, or common expressions to the condition.

Code 128 Settings

Code 128 is a high-density barcode, used for alphanumeric or numeric-only barcodes. It supports all 128 ASCII characters.

Note

Note

To create dynamic barcodes, "Metadata options" on page 849 must first have been set. Metadata fields are required to create the association between the dynamic data used in the print run and the barcode.



Use the following options to configure the output Barcode settings:

- **Position** group:
 - **Orientation:** Use the drop-down to select the orientation of the Barcode added to the page.
 - **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Barcode printed once per sheet rather than once per document page.

Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.

- **Left:** Enter the distance between the left margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
 - **Bottom:** Enter the distance between the bottom margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
- **Code 128 Properties** group:
 - **Height:** Enter the Barcode height in either metric (cm/mm), inch (in) or point (pt) values.

- **Module Width:** Specifies the width of the narrow bars. Changing this value to higher value will generally make the Barcode bigger.
- **Print human readable text:** Check to add a textual version of the Barcode data.
 - **Placement:** Use the drop-down to select whether to place the human readable text above or below the Barcode.
 - **Font name:** Use the drop-down to select the font with which to display the human readable text.
 - **Font size:** Enter a font size for the human readable text.
- **Text:** Enter the text used to generate the Barcode.
 -  **Add** button: Click to display a list of variable data that could be used for generating the Barcode. This includes metadata fields which must previously have been added in the [Metadata Options](#) (likely at the *Document Tags* level), as well as some information fields.
 - **Condition:** Enter the condition which determines whether or not the Barcode will be added to the document at print time. For details on how to create a conditional, see the Conditionals page.
 -  **Add** button: Click to display a list of metadata fields, information fields to add, or common expressions to the condition.

Code 39 Settings

Code 39 is a discrete, self-checking barcode that is also known as "Alpha39", "Code 3 of 9" (often abbreviated to "3 of 9"), "Code 3/9", "Type 39", "USS Code 39" and "USD-3".

Code 39 data should contain no more than 20 digits from within the following range: Numeric digits: (0-9), upper-case letters (A-Z), seven special characters (- . space \$ / + %) and the start/stop asterisk (*) character.

If the Extended character set is chosen, then lower-case letters (a-z) and other special ASCII characters can also be included.

Note

Note

To create dynamic barcodes, "Metadata options" on page 849 must first have been set. Metadata fields are required to create the association between the dynamic data used in the print run and the barcode.

Use the following options to configure the output Barcode settings:



- **Position** group:
 - **Orientation:** Use the drop-down to select the orientation of the Barcode added to the page.
 - **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Barcode printed once per sheet rather than once per document page.

Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.

- **Left:** Enter the distance between the left margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
 - **Bottom:** Enter the distance between the bottom margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
- **Code 39 Properties** group:
 - **Height:** Enter the Barcode height in either metric (cm/mm), inch (in) or point (pt) values.
 - **Use extended character set:** Check to use the Code 39 Extended character set. This extends the range of supported data to include the full ASCII character set.

This adds support for lower case letters (a-z) and the full range of ASCII punctuation and special characters.

- **Module Width:** Specifies the width of the narrow bars. Changing this value to higher value will generally make the Barcode bigger. The smallest Module Width is 0.19mm (high density).
- **Bar width ratio:** Set the Barcode bar width.
- **Checksum:** Use the drop-down to select how to deal with the Barcode checksum:
 - **Ignore:** Ignore checksum calculations.
 - **Auto:** Add a checksum character to the Barcode if the initial value does not validate. This is the default value.
 - **Check:** Verify the Barcode has a valid checksum.
 - **Add:** Calculate and add a checksum character to Barcode, regardless of current value.
- **Print human readable text:** Check to add a textual version of the Barcode data.
 - **Placement:** Use the drop-down to select whether to place the human readable text above or below the Barcode.
 - **Font name:** Use the drop-down to select the font with which to display the human readable text.
 - **Font size:** Enter a font size for the human readable text.
- **Text:** Enter the text used to generate the Barcode.
 -  **Add** button: Click to display a list of variable data that could be used for generating the Barcode.
This includes metadata fields which must previously have been added in the [Metadata Options](#) (likely at the *Document Tags* level), as well as some information fields.
- **Condition:** Enter the condition which determines whether or not the Barcode will be added to the document at print time. For details on how to create a conditional, see the [Conditionals](#) page.
 -  **Add** button: Click to display a list of metadata fields, information fields to add, or common expressions to the condition.

Data Matrix Settings

A Data Matrix barcode is a high-density, two-dimensional (2D) matrix barcode which supports encoded text, numbers, files and digital data.

Note

Note

To create dynamic barcodes, "Metadata options" on page 849 must first have been set. Metadata fields are required to create the association between the dynamic data used in the print run and the barcode.



Use the following options to configure the output Barcode settings:

- **Position** group:
 - **Orientation:** Use the drop-down to select the orientation of the Barcode added to the page.
 - **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Barcode printed once per sheet rather than once per document page.

Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.

- **Left:** Enter the distance between the left margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
- **Bottom:** Enter the distance between the bottom margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.

- **Datamatrix Properties** group:
 - **Module Width:** Specifies the width of the narrow bars. Changing this value to higher value will generally make the Barcode bigger.
 - **Encoding:** The data represented in the symbol can be compressed using one of the following algorithms:
 - **Auto:** Automatically detect the data content and encodes using the most appropriate method. This is the default option.
 - **ASCII:** is used to encode data that mainly contains ASCII alphanumeric characters (ASCII 0-127). Use where Barcode size is a concern and where the data is alphanumeric.
 - **Base 256:** used to encode 8-bit values.
 - **C40:** used for data that mainly consists of numbers and upper-case alphabetic letters.
 - **Text:** used for data that mainly consists of numbers and lower-case alphabetic letters.
 - **None:** Does not use any encoding.
 - **Format:** select the Barcode size format from the drop-down list .
- **Text:** Enter the text used to generate the Barcode.
 -  **Add** button: Click to display a list of variable data that could be used for generating the Barcode.
This includes metadata fields which must previously have been added in the [Metadata Options](#) (likely at the *Document Tags* level), as well as some information fields.
 - **Condition:** Enter the condition which determines whether or not the Barcode will be added to the document at print time. For details on how to create a conditional, see the [Conditionals](#) page.
 -  **Add** button: Click to display a list of metadata fields, information fields to add, or common expressions to the condition.

EAN-128 Settings

EAN128 is also known as "EAN/UCC 128", "UCC 128" and "GS1-128". This barcode type not only encodes data, but also provides a mechanism for defining the meaning (or format) of that data. It supports alphanumeric data and some predefined Function Codes. See the [Wikipedia](#)

[GS1-128 entry](#) for more information.

Note

Note

To create dynamic barcodes, "Metadata options" on page 849 must first have been set. Metadata fields are required to create the association between the dynamic data used in the print run and the barcode.

Use the following options to configure the output Barcode settings:

- **Position** group:
 - **Orientation:** Use the drop-down to select the orientation of the Barcode added to the page.
 - **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Barcode printed once per sheet rather than once per document page.

Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.



- **Left:** Enter the distance between the left margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
 - **Bottom:** Enter the distance between the bottom margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
- **EAN 128 Properties** group:
 - **Height:** Enter the Barcode height in either metric (cm/mm), inch (in) or point (pt) values.

- **Module Width:** Specifies the width of the narrow bars. Changing this value to higher value will generally make the Barcode bigger.
- **Check Digit marker:** This character is used as a placeholder for the check digit, which we be calculated at runtime. The character must be expressed in Hex.
- **Group separator:** This character is used to define group separation points. The character must be expressed in Hex.
- **Template:** Specify an optional Barcode "template".

Examples:

- `n13` defines a numeric field with exactly 13 digits.
- `n13+cd` defines a numeric field with exactly 13 digits plus a check digit.
- `an1-9` defines an alpha-numeric field with 1 to 9 characters.

Elements can be combined using the '+' symbol.

- **Print human readable text:** Check to add a textual version of the Barcode data.
 - **Placement:** Use the drop-down to select whether to place the human readable text above or below the Barcode.
 - **Font name:** Use the drop-down to select the font with which to display the human readable text.
 - **Font size:** Enter a font size for the human readable text.
- **Text:** Enter the text used to generate the Barcode.
 -  **Add** button: Click to display a list of variable data that could be used for generating the Barcode. This includes metadata fields which must previously have been added in the [Metadata Options](#) (likely at the *Document Tags* level), as well as some information fields.
 - **Condition:** Enter the condition which determines whether or not the Barcode will be added to the document at print time. For details on how to create a conditional, see the Conditionals page.
 -  **Add** button: Click to display a list of metadata fields, information fields to add, or common expressions to the condition.

EAN-13 Settings

EAN-13 barcodes are composed entirely of numerical data. The first 12 digits representing country/economic area, manufacturer and product codes + 1 following checksum digit.

Note

Note

To create dynamic barcodes, "Metadata options" on page 849 must first have been set. Metadata fields are required to create the association between the dynamic data used in the print run and the barcode.



Use the following options to configure the output Barcode settings:

- **Position** group:
 - **Orientation:** Use the drop-down to select the orientation of the Barcode added to the page.
 - **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Barcode printed once per sheet rather than once per document page.

Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.

- **Left:** Enter the distance between the left margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
 - **Bottom:** Enter the distance between the bottom margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
- **EAN 13 Properties** group:
 - **Height:** Enter the Barcode height in either metric (cm/mm), inch (in) or point (pt) values.
 - **Module Width:** Specifies the width of the narrow bars. Changing this value to higher value will generally make the Barcode bigger. The EAN-13 barcode employs a module width between 0.27mm and 0.66mm.

- **Checksum:** Use the drop-down to select how to deal with the Barcode checksum:
 - **Ignore:** Ignore checksum calculations.
 - **Auto:** Add a checksum character to the Barcode if the initial value does not validate. This is the default value.
 - **Check:** Verify the Barcode has a valid checksum.
 - **Add:** Calculate and add a checksum character to Barcode, regardless of current value.
- **Print human readable text:** Check to add a textual version of the Barcode data.
 - **Placement:** Use the drop-down to select whether to place the human readable text above or below the Barcode.
 - **Font name:** Use the drop-down to select the font with which to display the human readable text.
 - **Font size:** Enter a font size for the human readable text.
- **Text:** Enter the text used to generate the Barcode.
 -  **Add** button: Click to display a list of variable data that could be used for generating the Barcode. This includes metadata fields which must previously have been added in the [Metadata Options](#) (likely at the *Document Tags* level), as well as some information fields.
 - **Condition:** Enter the condition which determines whether or not the Barcode will be added to the document at print time. For details on how to create a conditional, see the Conditionals page.
 -  **Add** button: Click to display a list of metadata fields, information fields to add, or common expressions to the condition.

EAN-8 Settings

An EAN-8 barcodes are composed entirely of numerical data. It is comprised of 7 data digits containing the country/economic area code and an item reference code, with 1 following checksum digit.

Note

Note

To create dynamic barcodes, "Metadata options" on page 849 must first have been set. Metadata fields are required to create the association between the dynamic data used in the print run and the barcode.



Use the following options to configure the output Barcode settings:

- **Position** group:
 - **Orientation:** Use the drop-down to select the orientation of the Barcode added to the page.
 - **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Barcode printed once per sheet rather than once per document page.

Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.

- **Left:** Enter the distance between the left margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
 - **Bottom:** Enter the distance between the bottom margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
- **EAN 8 Properties:**
 - **Height:** Enter the Barcode height in either metric (cm/mm), inch (in) or point (pt) values.
 - **Module Width:** Specifies the width of the narrow bars. Changing this value to higher value will generally make the Barcode bigger. The EAN-8 barcode employs a module width between 0.27mm and 0.66mm.

- **Checksum:** Use the drop-down to select how to deal with the Barcode checksum:
 - **Ignore:** Ignore checksum calculations.
 - **Auto:** Add a checksum character to the Barcode if the initial value does not validate. This is the default value.
 - **Check:** Verify the Barcode has a valid checksum.
 - **Add:** Calculate and add a checksum character to Barcode, regardless of current value.
- **Print human readable text:** Check to add a textual version of the Barcode data.
 - **Placement:** Use the drop-down to select whether to place the human readable text above or below the Barcode.
 - **Font name:** Use the drop-down to select the font with which to display the human readable text.
 - **Font size:** Enter a font size for the human readable text.
- **Text:** Enter the text used to generate the Barcode.
 -  **Add** button: Click to display a list of variable data that could be used for generating the Barcode. This includes metadata fields which must previously have been added in the [Metadata Options](#) (likely at the *Document Tags* level), as well as some information fields.
 - **Condition:** Enter the condition which determines whether or not the Barcode will be added to the document at print time. For details on how to create a conditional, see the Conditionals page.
 -  **Add** button: Click to display a list of metadata fields, information fields to add, or common expressions to the condition.

Interleaved 2 of 5 Settings

Interleaved 2 of 5 barcodes are also known as "ITF" and "2/5 Interleaved". It is a numeric only barcode whose data must contain an even number of digits, as the barcode uses sequences of two digits interleaved with each other to create a single symbol. If the numeric data contains an odd number of digits, then a leading zero must be added to the beginning of the data.

Note

Note

To create dynamic barcodes, "Metadata options" on page 849 must first have been set. Metadata fields are required to create the association between the dynamic data used in the print run and the barcode.



Use the following options to configure the output Barcode settings:

- **Position** group:
 - **Orientation:** Use the drop-down to select the orientation of the Barcode added to the page.
 - **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Barcode printed once per sheet rather than once per document page.

Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.

- **Left:** Enter the distance between the left margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
 - **Bottom:** Enter the distance between the bottom margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
- **Interleaved 2 of 5 Properties** group:
 - **Height:** Enter the Barcode height in either metric (cm/mm), inch (in) or point (pt) values.
 - **Module Width:** Specifies the width of the narrow bars. Changing this value to higher value will generally make the Barcode bigger.
 - **Bar width ratio:** Set the Barcode bar width.

- **Print human readable text:** Check to add a textual version of the Barcode data.
 - **Placement:** Use the drop-down to select whether to place the human readable text above or below the Barcode.
 - **Font name:** Use the drop-down to select the font with which to display the human readable text.
 - **Font size:** Enter a font size for the human readable text.
- **Text:** Enter the text used to generate the Barcode.
 -  **Add** button: Click to display a list of variable data that could be used for generating the Barcode.
This includes metadata fields which must previously have been added in the [Metadata Options](#) (likely at the *Document Tags* level), as well as some information fields.
 - **Condition:** Enter the condition which determines whether or not the Barcode will be added to the document at print time. For details on how to create a conditional, see the [Conditionals](#) page.
 -  **Add** button: Click to display a list of metadata fields, information fields to add, or common expressions to the condition.

PDF417 Settings

PDF417 is a two-dimensional, multi-row Barcode. It is used for encoding large amounts of data, with hundreds or even thousands of characters. It encodes alphabetic text, numbers, binary files and actual data bytes.

Note

Note

To create dynamic barcodes, "Metadata options" on page 849 must first have been set. Metadata fields are required to create the association between the dynamic data used in the print run and the barcode.

Use the following options to configure the output Barcode settings:

- **Position** group:



- **Orientation:** Use the drop-down to select the orientation of the Barcode added to the page.
- **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Barcode printed once per sheet rather than once per document page.

Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.

- **Left:** Enter the distance between the left margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
 - **Bottom:** Enter the distance between the bottom margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
- **PDF417 Properties** group:
 - **Module Width:** Specifies the width of the narrow bars. Changing this value to higher value will generally make the Barcode bigger.
 - **Row height:** Defines the height of the bars for a single row, measured in pixels, points or metric.
 - **Width to height ratio:** Select the ratio of column width to row height.
 - **Mode:** Use the drop-down to set the compaction mode.
 - **Binary:** allows any byte value to be encoded.
 - **Text:** allows all printable ASCII characters to be end coded (ASCII values 32 to 126 and some additional control characters).
 - **Numeric:** more efficient mode for encoding numeric data.
 - **Auto:** Automatically detect the data content and encodes using the most appropriate method. This is the default option.
 - **Error Correction Level:** Enter the error correction level for the built-in error correction method based on Reed-Solomon algorithms. The error correction level is adjustable between level 0 (just error detection, without correction) and level 8

(maximum error correction). Recommended error correction levels are between level 2 and 5, but the optimal value depends on the amount of data, printing quality of the PDF417 symbol and decoding capabilities.

- **Rows:** A PDF417 bar code can have anywhere from 3 to 90 rows.
- **Columns:** The number of data columns can vary from 1 to 30.
- **Text:** Enter the text used to generate the Barcode.
 -  **Add** button: Click to display a list of variable data that could be used for generating the Barcode.
This includes metadata fields which must previously have been added in the [Metadata Options](#) (likely at the *Document Tags* level), as well as some information fields.
 - **Condition:** Enter the condition which determines whether or not the Barcode will be added to the document at print time. For details on how to create a conditional, see the [Conditionals](#) page.
 -  **Add** button: Click to display a list of metadata fields, information fields to add, or common expressions to the condition.

QR Code Settings

QR Code (Quick Response Code) is a 2D Barcode format that supports alphanumeric, numeric, byte/binary, and Kanji (Japanese-Chinese character) data.

Note

Note

To create dynamic barcodes, "Metadata options" on page 849 must first have been set. Metadata fields are required to create the association between the dynamic data used in the print run and the barcode.

Use the following options to configure the output Barcode settings:

- **Position group:**

- **Orientation:** Use the drop-down to select the orientation of the Barcode added to the page.
- **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Barcode printed once per sheet rather than once per document page.

Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.

- **Left:** Enter the distance between the left margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
- **Bottom:** Enter the distance between the bottom margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.

QRCode Properties group:

- **Size by:** Select size from the two options available:
 - **By area:** Connect will try to size the Barcode to fit the specified area by dynamically changing the module width to the **Size** selection. The lower module width limit is governed by the **Minimum module width** selection. Enter the sizes in either metric (cm/mm), inch (in), pixel (px) or point (pt) values.
 - **By module width:** Connect will try to size the Barcode to the module width of the characters. Large Barcode values will result in larger Barcode and vice versa. Enter the **Module width** in either metric (cm/mm), inch (in), pixel (px) or point (pt) values.
- **Encoding:** Define the encoding of the Barcode:
 - **Auto:** Automatically detect the data content and encodes using the most appropriate method. This is the default option.
 - **Numeric:** 7089 numerical characters.
 - **Alphanumeric:** 4296 alphanumerical characters.

- **Byte:** 2953 characters.
- **Kanji:** 1817 Japanese/Chinese characters.
- **Version:** Select the preferred QR code version (which sets the data length field) from the 40 available.

Note



The Encoding and Version fields work together to determine how many characters are encoded within a *length field*. The following table shows the number of bits in a length field, based upon the selections made:

Encoding	Ver. 1-9	Ver. 10-23	Ver. 27-40
Numeric	10	12	14
Alphanumeric	9	11	13
Byte	8	16	16
Kanji	8	10	12

- **Error Correction Level:** Part of the robustness of QR codes is their ability to sustain “damage” and continue to function even when a part of the QR code image is obscured, defaced or removed. A higher correction level duplicates data within the QR Code to allow for damaged areas. The higher the Error Correction Level, the larger the Barcode will be. The choices are (in order from lowest to highest): **Low**, **Medium**, **Quartile** and **High**.
- **Use ECI for encoding messages as bytes:** Selecting Extended Channel Interpretations (ECI) allows encoding multiple character sets (e.g. Arabic, Cyrillic, Greek, Hebrew) and other data interpretations, into one QR Code symbol.
- **Multi-part QR Code (structured append):** Select to append a QR Code symbol in a structured format.
 - **Part:** indicates the position of the QR Code symbol within the group of Structured Append symbols.
 - **of:** indicates how many Structured Append symbols exist.

Note

The Structured Append symbols Part number can never exceed the sum total of Structured Append symbols available (the "of" value). Thus selecting a Part number beyond the existing sum total will increase the sum total to the same value.

- **Use FNC1:** Check to enable Application Identifiers. These are often used to encode links to websites, or to encode production/batch details.
 - **Position:** Select between the two methods for encoding FNC1 characters within QR Codes:
 - **First Position** - uses the GS1 QR Code standard.
 - **Second Position** - uses the AIM QR Code standard. If this option is chosen then the appropriate Application Indicator will also need to be set.
 - **Application ID:** Enter the appropriate QR-Code Application Indicator in accordance with the specific industry or application specifications (as provided by AIM International).
- **Text:** Enter the text used to generate the Barcode.
 -  **Add** button: Click to display a list of variable data that could be used for generating the Barcode. This includes metadata fields which must previously have been added in the [Metadata Options](#) (likely at the *Document Tags* level), as well as some information fields.
- **Condition:** Enter the condition which determines whether or not the Barcode will be added to the document at print time. For details on how to create a conditional, see the Conditionals page.
 -  **Add** button: Click to display a list of metadata fields, information fields to add, or common expressions to the condition.

UPC-A Settings

The Universal Product Code (UPC-A) Barcode is widely used for tracking trade items in stores and at the point-of-sale. It consists of 12 numerical digits which are uniquely assigned to each trade item.

Note

Note

To create dynamic barcodes, "Metadata options" on page 849 must first have been set. Metadata fields are required to create the association between the dynamic data used in the print run and the barcode.

Use the following options to configure the output Barcode settings:

- **Position** group:
 - **Orientation:** Use the drop-down to select the orientation of the Barcode added to the page.
 - **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Barcode printed once per sheet rather than once per document page.



Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.

- **Left:** Enter the distance between the left margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
- **Bottom:** Enter the distance between the bottom margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.

UPC A Properties group:

- **Height:** Enter the Barcode height in either metric (cm/mm), inch (in) or point (pt) values.
- **Module Width:** Specifies the width of the narrow bars. Changing this value to higher value will generally make the Barcode bigger.
- **Checksum:** Use the drop-down to select how to deal with the Barcode checksum:

- **Ignore:** Ignore checksum calculations.
- **Auto:** Add a checksum character to the Barcode if the initial value does not validate. This is the default value.
- **Check:** Verify the Barcode has a valid checksum.
- **Add:** Calculate and add a checksum character to Barcode, regardless of current value.
- **Print human readable text:** Check to add a textual version of the Barcode data.
 - **Placement:** Use the drop-down to select whether to place the human readable text above or below the Barcode.
 - **Font name:** Use the drop-down to select the font with which to display the human readable text.
 - **Font size:** Enter a font size for the human readable text.
- **Text:** Enter the text used to generate the Barcode.
 -  **Add** button: Click to display a list of variable data that could be used for generating the Barcode. This includes metadata fields which must previously have been added in the [Metadata Options](#) (likely at the *Document Tags* level), as well as some information fields.
 - **Condition:** Enter the condition which determines whether or not the Barcode will be added to the document at print time. For details on how to create a conditional, see the Conditionals page.
 -  **Add** button: Click to display a list of metadata fields, information fields to add, or common expressions to the condition.

UPC-E Settings

The Universal Product Code (UPC-E) Barcode is widely used for tracking trade items in stores and at the point-of-sale. It consists of 6 numerical digits which are uniquely assigned to each trade item.

Note

Note

To create dynamic barcodes, "Metadata options" on page 849 must first have been set. Metadata fields are required to create the association between the dynamic data used in the print run and the barcode.

Use the following options to configure the output Barcode settings:

- **Position** group:
 - **Orientation:** Use the drop-down to select the orientation of the Barcode added to the page.
 - **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the Barcode printed once per sheet rather than once per document page.



Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.

- **Left:** Enter the distance between the left margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.
- **Bottom:** Enter the distance between the bottom margin of the page and the Barcode, in either metric (cm/mm), inch (in) or point (pt) values.

UPC A Properties group:

- **Height:** Enter the Barcode height in either metric (cm/mm), inch (in) or point (pt) values.
- **Module Width:** Specifies the width of the narrow bars. Changing this value to higher value will generally make the Barcode bigger.
- **Checksum:** Use the drop-down to select how to deal with the Barcode checksum:

- **Ignore:** Ignore checksum calculations.
- **Auto:** Add a checksum character to the Barcode if the initial value does not validate. This is the default value.
- **Check:** Verify the Barcode has a valid checksum.
- **Add:** Calculate and add a checksum character to Barcode, regardless of current value.
- **Print human readable text:** Check to add a textual version of the Barcode data.
 - **Placement:** Use the drop-down to select whether to place the human readable text above or below the Barcode.
 - **Font name:** Use the drop-down to select the font with which to display the human readable text.
 - **Font size:** Enter a font size for the human readable text.
- **Text:** Enter the text used to generate the Barcode.
 -  **Add** button: Click to display a list of variable data that could be used for generating the Barcode. This includes metadata fields which must previously have been added in the [Metadata Options](#) (likely at the *Document Tags* level), as well as some information fields.
 - **Condition:** Enter the condition which determines whether or not the Barcode will be added to the document at print time. For details on how to create a conditional, see the Conditionals page.
 -  **Add** button: Click to display a list of metadata fields, information fields to add, or common expressions to the condition.

OMR Mark Settings

The **Add OMR** dialog displays the properties of an OMR Mark that was added in the "Additional Content" on page 799 page.

These OMR marks differ from High Capacity Feeder (HCF) generated inserter marks. Those marks are specific to the inserter machine they were created for, whereas these additional OMR marks are completely independent and customizable. These custom OMR marks can be used to cater for inserter machines not currently support by a HCF, or they can be used for any non-inserter related post processing driven by OMR marks

- **Position group:**
 - **Orientation:** Use the drop-down to select the orientation of the OMR Mark added to the page.
 - **Page Side:** Select whether the OMR Mark will print on the front or back of page.
 - **Output once per sheet:** Option relates to [Imposition](#) (also known as N-Up) printing. Select this box to have the OMR Mark printed once per sheet rather than once per document page.

Note

If Imposition options such as auto-positioning and scaling were selected, these options won't apply to the Additional Content added to the physical N-Up sheet.

- **Left:** Enter the distance between the left margin of the page and the OMR Mark, in either metric (cm/mm), inch (in), pixel (px) or point (pt) values.
- **Bottom:** Enter the distance between the bottom margin of the page and the OMR Mark, in either metric (cm/mm), inch (in), pixel (px) or point (pt) values.
- **Options Tab:**
 - **Collation Level:** Choices are:
 - **Document:** Treats each document as a group and the group and match marks will be set based upon the start and end of a document.
 - **Document Set:** Treats each document set as a group and the group and match marks will be set based upon the start and end of a document set.
 - **Draw Hot Spots:** This adds a red rectangle around the location of each individual mark in the output, allowing easier checking of the OMR mark logic.
 - **Line Options group:**
 - **Line Thickness:** Sets the thickness of each OMR mark line.
 - **Line Length:** Sets the length of each OMR mark line.
 - **Line Spacing:** Determines how the spacing between each OMR mark line will be determined. The associated control beneath the combination box will be

enabled, based upon this selection.

- **Line Per Inch:** If **Line Spacing** is set to *Lines Per Inch* this option will be enabled. It defines how many lines will print per inch.
 - **Gap Distance:** If **Line Spacing** is set to *Gap Distance* this option will be enabled. It defines the size of the gap between lines.
i.e. the distance from the bottom of one OMR mark line to the top of the next.
 - **Line Distance:** If **Line Spacing** is set to *Line Distance* this option will be enabled. It defines the distance from the top one line to the top of the next.
- **Sequence Number Range** group: Allows selection of Start and Stop points for the wrapping page sequence number in a group.
For example, a range of 2-10 would cause the sequence numbers to iterate as follows: 2, 3, 4, 5, 6, 7, 8, 9, 10, 2, 3, 4 ...

Note

The Sequence Number iterates per page within a group and is used to identify missing pages in a group.

- **Start:** The starting point for the range
 - **Stop:** The end point of the range
 - **Start number:** The number to start from (from within the selected range).
- **Match Number Range** group: Allows selection of Start and Stop points for the wrapping match number for a group.
For example, a range of 1-6, with a Start number of 2 would cause the matched numbers to be as follows: 2, 3, 4, 5, 6, 2, 3, 4, 5, 6, 2, ...

Note

The Match Number iterates per group and is used to identify missing groups

- **Start:** Start number
- **Stop:** Stop number
- **Condition:** Enter the condition which determines whether or not the OMR Mark will be added to the document at print time.
For details on how to create a conditional, see the Conditionals page.
- **OMR Marks Tab:**
 - **#:** OMR Mark number (display only).
 - **Type:** Type of OMR Mark (display only).
 - **Value:** OMR Mark Value. These can be selected and altered for Sequence, Match and Parity marks, as described below.
 - **Add:** Add an OMR Mark entry to the table.
Choices are between:
 - **On:** This represents a mark that is always printed
 - **Off:** This represents a mark that is never printed.
i.e. it pads the marks out with an empty position
 - **Group Start:** This represents a mark that is printed on the first page of a group
 - **Group End:** This represents a mark that is printed on the last page of a group





Note

In a single page group both Group Start and End marks will print if defined since the page is both the start and end of the group.

- **Sequence:** This represents a mark that is printed when the specified bit is set in the sequence number of the page.
For example, if the bit for the mark is set to 2 and the sequence number for the page is 5 then it will not print since the value 5 consists of the bits 1 and 4.
Use the drop down box to select the entry.
- **Match:** This represents a mark that is printed when the specified bit is set in the match number of the group.
For example, if the bit for the mark is set to 2 and the match number for the group is 3 then it will print since 3 consists of the bits 1 and 2.
Use the drop down box to select the entry.

Note

The match number is the same for all pages in a group

- **Parity:** This mark prints in order to maintain the parity of the number of lines printed on the page. If set to *Even* then it will print if the total count of the other printed marks in the printed is odd.
For example, by printing the parity mark it will create an even number of marks on the page. And vice versa with *Odd* parity - the parity mark will print if the total number of other printed marks on the page is even in order to keep the overall count odd.
Use the drop down box to select the entry.
- **Conditional:** Enter the condition which determines whether or not this OMR Mark will be added to the document at print time.
For details on how to create a conditional, see the Conditionals page
-  **Delete:** Delete an entry from the table
-  **Move up:** Move a entry up the table
-  **Move down:** Move a entry down the table
-  **Edit:** Edit a **Conditional** entry within the table.

Tip

You can also double click a **Conditional** entry within the table to edit it.

PDF Options

The **PDF Options** page is shown only when a PDF Print output type is selected in the [Print Options](#) dialog. It is used to select PDF specific options.

• PDF Options Group

- **PDF Type:** Use the drop-down to specify which format the PDF should be generated in. These options are standard PDF, archive format PDF (PDF/A-1b), graphics format PDF (PDF-X4) and variable data printing format PDF (PDF-VT).

- **Embed standard fonts:** Click to embed the 14 standard system fonts within the PDF output. This increases the output filesize but makes the PDF output truly portable. Such PDFs print as displayed on screen, regardless of whether the 14 standard fonts are present on the target printing system or not.

Note

This box is ignored for PDF/A and PDF-X4 output, as fonts are always embedded in those output types.

- **Pass-through PDF resources:** Click to have PDF resources used *as-is*, without any additional processing. This guarantees the fidelity of any PDF graphics used within the template will be retained in the output.

Note

Connect tries to write content in the best way possible, depending on the chosen output format and optimization settings. Selecting *PDF pass-through* means the output will be less optimized, which typically produces somewhat larger files.

- **Add Digital Signature Group:** Check to enable the integration of a digital signature into the PDF.













A digital signature identifies the person signing a document, similarly to a conventional handwritten signature. Unlike a handwritten signature, a digital signature is difficult to forge as it contains encrypted information which is unique to the signer and which can be password protected and verifiable.

- **All Keystores:**

Here you can choose from existing digital signatures, or select new ones.

- **Name:** The user-defined name of the keystore.
- **File:** The file path and name to the keystore file.

This is where you select keystore values.

-  **New**: Click to open the [Key Store](#) dialog to add a new keystore to the list.
-  **Duplicate**: Click to make a copy of the currently selected keystore.
-  **Edit**: Click to edit the currently selected keystore in the [Key Store](#) dialog.
-  **Delete**: Click to delete the currently selected keystore.
-  **Move Up**: Click to move the currently selected keystore up.
-  **Move Down**: Click to move the currently selected keystore down.
- **All Signatures**: Displays a list of signatures to add to the PDF output.
 - **Name**: The user-defined name of the signature.
 - **File**: The file path and name to the signature file.
 - **Alias**: The user-defined alias for the signature.
 -  **New**: Click to open the [PDF Signature](#) dialog to add a new signature to the list.
 -  **Duplicate**: Click to make a copy of the currently selected signature.
 -  **Edit**: Click to edit the currently selected signature in the [PDF Signature](#) dialog.
 -  **Delete**: Click to delete the currently selected signature.
 -  **Move Up**: Click to move the currently selected signature up.
 -  **Move Down**: Click to move the currently selected signature down.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.
- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print

Options" on page 851 page have been completed or not.

- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Keystore

The security certificate **Keystore** dialog appears when adding or editing a keystore from the "PDF Options" on page 868 page.

This dialog allows you to select a keystore with a private key.
The keystores currently supported by Connect are:

- JKS (Java Key Store) format.
- PKCS#12
- PKCS#11

Note

PKCS#11 requires an extra plug-in not included in the PlanetPress Connect installation.

These are the options available in this dialog:

- **Name:** Enter a name for the keystore to describe it within Connect.
- **File:** Enter the path to the keystore file, or use the Browse button to locate the file.
- **Keystore properties group:**
 - **Type:** Use the drop-down to select the appropriate type of the keystore format the file is: JKS, PKCS11, PKCS12.
 - **Provider:** Enter the provider of the keystore.
 - "SUN" for JKS
 - "SunJSSE" for PKCS#12
 - "IAIK PKCS#11:1" for PKCS#11
 - **Password:** Type in the password that secures the keystore, if the keystore is password protected.

- **Repeat Password:** Re-type in the password that secures the keystore. Once this is done the two Password entry boxes will no longer have the red cross icon (indicating incomplete or unselected) flag beside them.
- **Properties file group:**
 - **File:** Load optional keystore properties file. Could be used to store the password in a file.

PDF Signature

The **PDF Signature** dialog appears when adding or editing a signature from the "PDF Options" on page 868 page.

- **Name:** Enter a name that describes the signature entry.
- **Keystore:** Use the drop-down to select which keystore the signature is pulled from. These keystores are set in the "Keystore" on page 872 dialog, called from the "PDF Options" on page 868 page.
- **Signature Properties group:** These are optional Metadata fields associated with the signature, which can be omitted.
 - **Location:** The CPU host name or physical location of the signing.
 - **Reason:** Records the reason for the signing.
 - **Contact:** Information to enable a recipient to contact the signer to verify the signature. For example: a phone number.
 - **Handler:** The PDF reader plugin used to interpret the signature data. It should be left at its default setting (Adobe.PPKLite) unless time-stamping is desired, in which case "Adobe.PPKMS" is likely the best option.
- **Key group:** Refers to a key from the keystore.
 - **Alias:** The user-friendly name of the key
 - **Password:** Enter the password for the key (the same password as was entered in [Key Store](#)).
 - **Repeat Password:** Re-enter the password for the key (same as previous).
- **Apply Time Stamping Authentication group:** Check to enable time stamping authentication.

Note

Not available for signatures set to use Adobe.PPKLite Handler.

- **URL:** Select the Time Stamp Authority (TSA) URL address.
- **Account:** Account name specific to the TSA server chosen.
- **Password:** Password specific to the TSA server chosen.
- **Repeat Password:** Repeat of password.
- **Visible Signature group:** Check to add a visible signature to the PDF file.
 - **X:** Enter the horizontal distance between the left side of the page and the left side of the signature, in points (pt).
 - **Y:** Enter the vertical distance between the top of the page and the top of the signature, in points (pt).
 - **Width:** Enter the desired width of the signature, in points (pt).
 - **Height:** Enter the desired height of the signature, in points (pt).

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.
- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Job Creation Presets

The **File>Print Presets>Job Creation Settings ...** dialog displays a list of available presets and a summary of their settings. This dialog can be used to create new Presets or to edit and update existing Presets. Presets, however, cannot be deleted or renamed from within this dialog. That must be done manually.

The Presets are all stored as individual files, using the Preset name and a "OL-jobpreset" file extension.

The Presets can be found in the following folder:

C:\Users\[username]\Connect\workspace\configurations\JobCreationConfig

Where [username] needs to be replaced with your own Windows username.

Dialog Interface

- **Data Mapping Configuration:** Use the drop-down to select which data mapping configuration this job creation preset will be based on. The data mapping configuration's model is used for field names in sorting, etc.
- **Configuration Name:** Use the drop-down to select the presets saved in the default location. Click the Gear icon for more options:
 - Click the **Reload** option to look for new presets.
 - Click the **Import Configuration...** option to import one or more Job Presets using a Browse dialog.
- **Properties:** Displays a summary of the settings for this Job Creation Preset.
 - **Has Custom Job Creation Options:** Indicates if any job creation settings have been added. Becomes Yes if any setting in any of the below windows have been added:
 - **Has Data Selection Filter:** Becomes Yes if [Data Filtering Options](#) are set.
 - **Has Sorting:** Becomes Yes if any [Sorting Options](#) are set.
 - **Has Grouping:** Becomes Yes if grouping options are set in the [Grouping and Splitting Options](#).
 - **Page Count Splitting:** Becomes Yes if page count splitting is used in the [Grouping and Splitting Options](#).

- **Slip Sheets:** Becomes Yes if a slip sheet is set in the [Grouping and Splitting Options](#).
- **Options Group:** These options are checked, or not, depending on the selected preset chosen in the Configuration name.
 - **Use Grouping:** Check to activate the [Grouping and Splitting Options](#) page of the wizard.
 - **Apply filtering and sorting to record selection:** Check to activate the [Data Filtering Options](#) page of the wizard.
 - **Include Metadata (PDF and AFP only):** Check to activate the [Metadata Options](#) page of the wizard.
 - **Override Finishing Options:** Check to activate the [Finishing Options](#).
- **Next:** Click to go to the next page of the Job Creation Wizard, [Data Filtering Options](#)
- **Finish:** At any point during the wizard, click to save the current configurations, whatever page you are on.
- **Cancel:** At any point during the wizard, click to exit the wizard without saving changes.

Finishing Options






Use the **Finishing Options** page to force the use of specific printer finishing options, rather than using any finishing options that might have been set in the Template's Media and Section options.

These settings are only applied when producing print output. They do not modify the original finishing options in either the Section or the Template.

Finishing settings can be set on various levels of Job Creation, such as Document Sets and Job Segments. This allows you, for example, to staple Document Sets whilst punching holes in Job Segments.

You can also have multiple finishing settings at the same level.

- **Finishing Options Table:** This table, on the left of the Wizard page, allows you to add or remove sections to apply the Finishing options to. It contains the **Section** for which the settings are to be applied (*Section, Document, Document Set or Job Segment*), plus the **Finishing** options that are to be applied, and any **Rule** determining the conditions under which the Finishing settings are to be applied. To set the details, use the following options:

-  **Import Finishing:** Imports the existing settings from a Connect Template. The Template could be the current template (*Import from currently template*), or some other Template (*Import from template ...*).
-  **Add:** Adds a new section to apply Finishing options to. The default is to add a new *Section*, but these can be changed to *Document*, *Document Set* or *Job Segment* types, thereafter.
You can have multiple selections of each type, all with different criteria determining when the Finishing options are to be applied.
-  **Delete:** Removes the selected Section(s) from the table.
-  **Move Up / Move Down:** Moves the selected Section(s) up or down within the table.
-  **Edit ... :** Brings up the Rule Editor Dialog, which can be used to construct the rule (s) that determines whether this conditional Finishing option is to be applied or not.
- **Binding group:**
 - **Style:** What type of Binding to request on the printer. This includes *Stapled*, *Glued*, *Stitched*, *Ring*, *Comb*, *Coil*, amongst other options..
 - **Side:** Sets the side of the paper that the Binding is to occur. This includes both the top and the bottom of the paper,
 - **Location:** Sets where the binding is to occur, if applicable.
The selections available here will be dependent upon the selection made in the Binding **Style**. Only Stapled and Stitched bindings have a **Location** option available to them.
 - **Angle:** Set Stapling or Stitching binding either horizontally, vertically, or at an angle (as supported by printer).
 - **Item count:** Select the amount of Staples or Stitches to use. The choice is between the default amount or selecting a specific number using the Count option.

Tip

The options available to you in reality at print time will be printer dependent, so you will need to know the capabilities of your printer, or leave the value set to Default.

- **Area:** The area where the binding can be applied.
- **Hole making group:**
Hole making options are available only to *Ring*, *Comb* (wire and plastic) and *Coil* Binding **Styles**. The selections will need to be made at run-time based upon the types of binding options available that the printer supports.
 - **Number of holes:** The number of holes to punch for the selected Binding option.
 - **Pattern Catalog ID:** The Catalog ID of the selected Binding option.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.
- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Data Filtering Options

The **Data Filtering Options** page is used to filter records and prevent them from being printed. Conditions are evaluated on each record.

Data Selection Filter

- **Grouping:** Displays the type of line, either a Rule or a rule Grouping. The root of each group of rule is a drop-down selector that defines how the rules inside the grouping work together, which is either to make *any of the rules* or *all of the rules* have to be true for the group to be true.
- **Field:** Use the drop-down to select the field on which to make the comparison.
- **Operator:** Use the drop-down to select the comparison operator for the condition.

- **Value:** Type in a value for the comparison.
- **Add:** Click to add a new line to list. Different options are available in this menu, such as filtering by field, media and finishing properties, or document length.
- **Add a new nested rule group:** Click to add a new grouping at the current level.
- **Delete:** Click to delete the currently selected rule or group. **Note:** deleting a group deletes all rules under it, and this action cannot be undone.
- **Group selected rules as nested rules:** Click to create a group with the currently selected rules.
- **Merge selected rules/ruleset to parent rules:** Click to move the currently selected rule (s) to the parent group.

Preview

This box displays a textual representation of the conditions set in the data filtering.

Sorting Options

The **Sorting Options** page is used to sort the records in the output. Sorting is done from the top to the bottom, one after the other.

Sorting Settings

- **Use standard sort:** Sort using the fields below:
 - **Field Name:** Use the drop-down to select which field to sort on.
 - **Order:** Use the drop-down to choose Ascending or Descending.
 - **Add:** Click to add a new row to the sort list. The list that appears contains all the fields in the Data Model, as well as a special <Document Length> option which is used to sort by the number of pages in each document.
 - **Delete:** Click to delete the currently selected row in the list.
 - **Move up:** Click to move the currently selected row up in the list.
 - **Move down:** Click to move the currently selected row down in the list.
- **Use external sort:** Sort the records using some external sorting software. A CSV file is exported, sorted by the external application and the sorted CSV file is returned and integrated, with the records now sorted according to the new order in the CSV file.

Warning







External Sort commands must return a non-zero error code if an error occurs.

An external sort command could easily fail part way through processing and generate only a partial output file. Without receiving a return code from the external sort process, PlanetPress Connect cannot know if the sort has successfully completed or not. Thus the sort program must generate a return code, with a code of zero ('0') indicating success, and all non-zero results indicating failure.

- **Command:** Enter either:
 - The full path to the executable that will sort the CSV file.
 - A valid Windows command line instruction to sort the records.
This instruction should do the following:
 1. Do some processing of the input CSV file which PlanetPress Connect will pass through in the position of the `${input}` placeholder.
 2. Generate an output file that contains the sorted data and must be named according the file name PlanetPress Connect will pass through in the position of the `${output}` placeholder

For example: `cmd /C sort /R ${input} ${output}`
This would reverse the order of the `${input}` file, and sent the output to the `${output}` file.
- **Separator:** Enter the field separator used in the CSV file, such as a comma (,), pipe (|), semicolon (;), etc.
- **Quote Character:** Enter the quoting character that wraps around any field that contains the separator.
- **Escape Character:** Enter the character use to escape the Quote character if it appears in the field value.
- **Line Ending:** Use the drop-down to select which line ending to use. The selections are: Windows *Carriage Return/Line Feed* combination (CRLF), Linux *Line Feed* (LF) or Apple Macintosh *Carriage Return* (CR).
- **Character Set:** Use the drop-down to select which character set to use when encoding the CSV file. This always defaults to UTF-8, as this caters for all possible

characters, is relatively compact (in terms of Unicode character sets) and is compatible with standard ASCII.

- **Exported sort data** group:
 - **First row of sort data has field names** checkbox: select to have field names placed on the first line of the exported CSV file.
 - **Fields to export:** Lists the fields to export in the CSV file. The buttons to the right of the table provide the following functionality:
 -  Click to select from available datafields. The Field Selection dialog will appear, which allows selection of one or several fields from those available.
 -  Click to remove a field from the list.
 -  Click to move fields up or down in the order of output.
 - **Record ID Field:** The Record ID field is a database *Primary Key* field, which is automatically added to the exported data file. The Record ID field name defaults to *RecID*, but can be changed here as desired.
- **Processing returns data** group: This allows the external sort application to introduce new data for each record. This data can be embedded in the metadata and used as the source for additional content within PlanetPress Connect. An example usage would be generating a postcode or postal barcode data from address details, making it available for use in PlanetPress Connect.
 - **Processing returns data** checkbox: Select this if the sort processing will be returning data. This activates the whole optional **Processing returns data** subsection.
 - **First row of return data has field names** checkbox: select to have field names placed on the first line of the returning datafile.
 - **Return Fields:** Lists the fields available in the selected data mapping configuration that can be used to sort the records. Fields can be added or removed by use of the add datafield () and remove datafield () buttons, or re-arranged with the arrow buttons (). Field names can be altered by selecting the field in the table, and editing the name. Fields can be made available to PlanetPress Connect via the "Include in meta data" checkbox. Click the checkbox beside the field name to make that datafield available as meta data.

- **Record ID Field** selection box: Select which return field is to be the Record ID field.
- **Sorting by** selection box: Select whether the sorting will be by the returned sort order or whether it is to be sorted on a selected datafield.
 - **Sequence Field** selection box: Select the datafield to be sorted on, if such was chosen in the **Sorting by** entry.

Grouping Options

The **Grouping options** page separates the job output into multiple blocks that can then be physically separated using split sheets in the printer.

- **Grouping Tabs:** Jobs can be grouped at three different levels. The three groups each have their own tab, and are as follows:
 - **Job** Grouping Fields
 - **Job Segment** Grouping Fields
 - **Document Set** Grouping Fields

The Fields available to be used for Grouping are contained within the **Available Fields** box.

Any Fields that you want to use for Grouping need to be added to the **Selected Fields** box via the arrows found between the two boxes.

Simply select the Field(s) you want to move and then click the appropriate arrow.

Any fields that you decide don't need to be used in Grouping can be returned to the Available Fields box in the same fashion.



Once a field is added to the **Available Fields** box, the *Sorting Option* can be selected by clicking in the "**Sorting Option**" column, and selecting the appropriate option. The selection can be made between Ascending or Descending order, or Unsorted.

- **Page Break Grouping:** Check this check box to enable page break grouping, which separates Documents into different groups, based on the number of pages they contain. For example, enabling the *Document Set Grouping* Level and creating a page range from 1-5 and 6 to Largest, will create two Document Set groups. The first will contain all Documents of 1 to 5 pages length, the second will contain any document of 6 or more

pages.

Note

Page Break Grouping works only on Document page counts.

- **Grouping Level:** Use the drop-down to select which grouping level to use, between **Job**, **Job Segment** or **Document Set**. Only one grouping level can be selected.
- **Grouping list:** Add  (or remove ) entries to this list to create new groups based upon the number of pages in the level selected above. All groups must be contiguous from 1-to-Largest and they must not contain any gaps.
 - **Range Name:** Enter a name identifying the range. It must be unique, but otherwise bears no impact on the range feature.
 - **From:** Enter the starting page number of the range. The first range must start with 1, all other ranges must be contiguous (the "From" range must be one higher than the previous "To" value).
 - **To:** Enter the last page number for the range. The last range must end with a selection of "Largest".
- **Generate page break ranges in reverse order:** Reverses the order of the groups created.

By default, grouping is done from smallest to largest. Checking this option instead creates groups from largest to smallest.
- **Generate page break range groups after normal grouping:** Check this option to first group using the levels above, following which page break grouping are applied. This creates two different levels of grouping, applied in order.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way

through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.

- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Metadata options

The **Metadata Options** page defines metadata tags that will be added to the output file when producing PDF and AFP output in the "Output Creation Settings" on the facing page.

Metadata tags are ignored in all other output types, except when they are associated with "Additional Content" on page 799 in the Print Wizard.

The tags can be added to any of these levels, as indicated by the tabs on top: **Job**, **Job Segment**, **Document**, **Document Set**, and **Page Tags**.

In each of these levels, a list of tags is available:

- **Always create meta data for this level even when fields are selected:** Select to create a blank meta data entry if no fields are selected. Done to ensure that a metadata store is always available, if required.
- **Tag Name:** Name of the metadata tag added to this level. Once a tag has been added, its name can be edited by double-clicking on the Tag Name.
- **Source Type:** Displays the type of field being used - either Text or Data Field.
- **Source:** For Data Fields only. The Field name from the data mapping configuration whose value will be used for this tag.
- **+ Add Field:** Click to add a new tag to the current level. The Field Selection dialog appears. Select either **Add field meta data** or **Add text meta data**. When adding field meta data select a field name from the Field List and click OK to add it as a tag of the same name.
- **- Delete Field:** Click to delete the currently selected tag.
- **↑ Move Up:** Click to move the currently selected tag one position up.
- **↓ Move Down:** Click to move the currently selected tag one position down.

Output Creation Settings

The **File>Print Presets>Output Creation Settings ...** dialog displays a list of available presets and a summary of their settings. This dialog can be used to create new Presets or to edit and update existing Presets. Presets, however, cannot be deleted or renamed from within this dialog. That must be done manually.

The Presets are all stored as individual files, using the Preset name and a "OL-outputpreset" file extension.




The Presets can be found in the following folder:

C:\Users\[username]\Connect\workspace\configurations\OutputCreationConfig


Where [username] needs to be replaced with your own Windows username.

Dialog Interface

Configuration Selection section:

- **Configuration Name:** Use the drop-down to select the presets saved in the default location.
 -  Click the **Settings** button for more options:
 -  Click the **Reload** option to look for new presets.
 -  Click the **Import Configuration...** option to import one or more Output Presets using a Browse dialog.
- **Properties:** Displays a summary of the settings for this Output Creation Preset.
 - **Output Type:** Displays the print technology used, as defined in the [Print Options](#)
 - **Inserter:** Indicates whether Inserter Marks have been added in the [Inserter Marks](#) dialog. Expand to see which High Capacity Feeder (HCF) model is loaded.
 - **Imposition:** Indicates if Imposition has been set in the "Imposition Options" on page 858 dialog. Expand to see the specific imposition settings.
 - **Has custom printer settings:** Indicates if custom printer settings have been set in the [Printer Settings](#) dialog. Expand to see the list of settings.
 - **Output to:** Indicates where the output will be done, either to a file or a printer.
 - **Has Custom Finishing:** Indicates that the output creation settings contain custom finishing overrides.

Optional Job Creation Configuration section:

- **Configuration Name:** Use the drop-down to select a preexisting Job Preset that can then be used as the source of Metadata and Grouping information in the output file mask dialog.
 Click the Reload button to refresh the list of available configurations.

Click Next in this dialog to see the [Print Options](#) window where output creation settings are selected.

Print Options

The **Print Options** page is the first page of both the **Advanced Print Wizard** and the [Output Creation Settings](#) Preset .

This page is the most important of the Advanced Print Wizard.

The other pages that appear throughout the Wizard are determined by the selections made on this page.

The choices can be broken down as follows:


- **Printer** group:
 - **Model:** Use the drop-down to select the printer language / output type that will be generated.
Connect output options cover a range of industry standard print output types. These include PCL, PDF and PostScript (including PPML, VIPP and VPS variants), with a range of quality settings available.

Note

By default, Connect displays only the PDF output option, but other print output types can be added to the Printer Model drop down list via the Settings button



For more information on how to do this, see "Adding print output models to the Print Wizard" on page 960.

- **Output Options** group:
 - **Output Local** checkbox: Select to have the output created using the local Print Server, instead of the Connect Server (see "The Connect server" on page 95 for an explanation).
 - **Output Type** choices:
 - **Prompt for file name**: Select to output to a local file on the hard drive. When this option is selected, no other configuration is necessary. A Save As dialog will appear to allow selection of the folder and filename.
 - **Directory** : Select to output to a local folder on the machine. Selecting this will open the **Directory Options** sub-group, which has these options:
 - **Job Output Mask**: The name of the file that will output. You could write the Job Output Mask directly into this edit box (for a list of available variables see "Print output variables" on page 961), or you could create a Mask via the Options  button. This opens the custom dialog: Job Output Mask Dialog. The Job Output Mask may contain (dynamic) folder names, for example: `${document.metadata['Country']}\${template}`. The evaluated value of the Job Output Mask is taken as a path **relative** to the folder specified by the Job Output Folder (the next option in this dialog). The Job Output Folder must exist, but folders specified in the Job Output Mask will get created if they don't exist.
 - **Job Output Folder**: The path on the disk where the file is produced. Please note that the folder must exist, or output will fail when produced through the server.
 - **LPR Queue**: Select to send the print job to an LPR queue. It is assumed that the print technology is supported by the system receiving the LPR job.
 - **Local Printer**: The IP or host name of the printer or machine where the LPD is installed and will receive
 - **Queue Name**: The queue name that will accept the job on the LPD. Default is generally "auto".
 - **Job Owner Name**: Optional entry for adding the name of the job owner.

- **Job Name:** The name of the output file. You can use `#{template}` as a variable for the name of the Designer Template used to generate the output.
- **Windows Printer:** Select to send the Print Job to a Printer Queue. The job is rendered as a PDF before being printed through the Windows driver.
 - **Windows Printer:** Use the drop-down to select the windows printer queue where the job will be sent.
 - **Job Owner Name:** Optional entry for adding the name of the job owner.
 - **Job Name:** The name of the output file. You can use `#{template}` as a variable for the name of the Designer Template used to generate the output.
- **PDF Rendering Options (PDF output only):**
 - **Auto-rotate and center:** Check to automatically select the page orientation that best matches the content and paper.
 - **Choose paper source by page size:** Check to use the PDF page size to determine the output tray rather than the page setup option. This option is useful for printing PDFs that contain multiple page sizes on printers that have different-sized output trays.
 - **Scale:**
 - **None:** Select to not scale any page, whether it fits or not.
 - **Expand to printable area:** Select to expand any page to fit the page area. Pages larger than the paper size are not resized.
 - **Shrink to printable area:** Select to shrink any page to fit the page area. Pages smaller than the paper size are not resized.
- **Production Options:**
 - **Booklet Imposition** checkbox: Check to tell the printer to generate a booklet for the print output. Booklet options are set in the "Booklet Options" on page 857 page. This option is unselected by default unless selected in the Designer "Print section properties" on page 720.
 - **Cut and Stack Imposition** checkbox: Check to enable Cut & Stack Imposition, which is set in the "Imposition Options" on page 858 page.

- **Add Inserter marks** checkbox: Check to enable inserter mark functionality, which is set in the "Inserter Options" on page 865 page.
- **Override Finishing options** checkbox: Check to configure custom "Finishing Options" on page 841, such as binding.
- **Print virtual stationery** checkbox: Check to enable virtual stationery in the output.
- **Use grouping** checkbox: Check to configure grouping of output into jobs, job segments or document sets. See "Grouping Options" on page 847.
- **Include meta data** checkbox: Check to add meta data to the output. This can be done at Job, Job Segment, Document, Document Set and Page level. See "Metadata options" on page 849.
- **Separation**: Check to activate the "Separation options" on page 856 page of the wizard.
- **Add additional content** checkbox: Check to activate the "Additional Content" on page 799 page of the wizard.
- **Records** group:
 - **Record Range**: Allows selection of a range of records or a custom selection. You can specific individual records separated by semi-colons (;) or ranges using dashes.
For example: 2;4;6-10 would print pages 2, 4, 6, 7, 8, 9 and 10.
- **Copies** section:
 - **Copies**: Enter the number of copies to print, of each record.
 - **Collate**: When printing multiple copies you can check this checkbox to have the record copies printed together.
For example in a three record job the records would print out as 1-1-2-2-3-3, rather than 1-2-3-1-2-3.
- **Pure Color Thresholds** group:

This section is valid for PCL only. It applies to elements within the record that are shades of gray, rather than black or white.

 - **Black Threshold Percentage**: The percentage of shading at which the element will appear as full black, rather than dark gray.
 - **White Threshold Percentage**: The percentage at which the element will appear as full white, rather than light gray.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.
- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Printer Settings

The **Printer Settings** page defines options on the printer. It is available for PostScript (and the VIPP and VPS variants of PostScript) only.

- **Map media by** options: Select from following choices:
 - **Media Attribute** displays all Media details, except the Tray selection.
 - **Tray** displays just the Media name and Tray selections.
 - **Both** displays all Media details.
- Tray selection columns:
 - **Media**: Lists the Media name, as defined in the template.
 - **Tray**: Use the drop-down to select in which tray to send any page using the media.
 - **Position**: Enter a MediaPosition option on the printer to define the media to use.
 - **Weight**: Enter a weight for the paper.
 - **Type**: Use the drop-down to select which type of stock to use on the printer.
 - **Color**: Use the drop-down to select which color the paper should be on the printer.

Separation options

The **Separation Options** page defines how to separate the jobs using subsets, slip sheets, or jogging.

- **Sheet Count Splitting** group.

This group allows for the splitting of output based upon a pre-determined number of pages

- **Split:** Use the drop-down to select how to split.
 - **None:** Select to ignore sheet count splitting entirely.
 - **At exactly:** Select to create a split at a specific sheet number.
- **Every:** Enter the number of sheets at which to split the output.

- **Separation Settings** group.

This setting is only available if no Sheet Count Split were specified.

- **Separation:** Use the drop-down to select when a job separation occurs, which is either **None** (no separation) or at the **Job**, **Job Segment**, **Document** or **Document Set** level.

- **Slip Sheets** group

- **Add slip sheet:** Use the drop-down to select whether to add a slip sheet before or after a specific separation, or whether to use none.
- **Every:** Use the drop-down to select at which separation to add a slip sheet, at the **Job**, **Job Segment**, **Document** or **Document Set** level.
- **Media Size:** Use the drop-down to select the media size of the slip sheet.
If a custom Media Size was chosen:
 - **Width:** enter slip sheet page width.
 - **Height:** enter slip sheet page height.

- **Jog** group

- **Jog after every:** Use the drop-down to select when to jog the printer, which is either **None** (no forced jogging) or at the **Job**, **Job Segment**, **Document** or **Document Set** level.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.
- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Booklet Options

The **Booklet Options** page defines how to generate booklets in the output. It is used in conjunction with [Imposition](#) settings, which will appear after the Booklet entries have been made.

This page includes a handy illustration that displays how the final binding would look, based upon the current selections.

Options:

- **Configuration**: Use the drop-down to select the type of binding to use:
 - **Saddle Binding**: This binding places all the pages in a stack, binds the middle and folds the stack as one.
 - **Perfect Binding**: This binding type is often used for books. Pages are folded in the middle and then set side by side. The pages are then bound along the folded "spine".
 - **1 up Perfect Binding**: This binding does not contain any folding. The pages are lined up side by side and bound along one edge.
- **Booklet Binding Edge**: Use the drop-down to select the side on which to bind the booklet.

Optional **Cover Page** selections are available to Saddle Binding only.

- **Cover Page** checkbox: Check to enable cover pages to be created with the options below:
 - **Media** selections:
 - **Cover Media Size**: Use the drop-down to select the media size for the cover page, or use a Custom size and select **Width** and **Height** values.
 - **Front Cover** selections:
 - **Blank**: Select to add no data to the front cover.
 - **First page on outside and second page on inside**: Select to use the first 2 pages as the inside and outside of the front cover.
 - **Back Cover** selections:
 - **Blank**: Select to add no data to the back cover.
 - **Last two pages on inside and outside**: Select to use the final 2 pages as the inside and outside of the back cover.

Imposition Options

Imposition refers to the printing of multiple pages on a single sheet. This is also known as N-Up printing.

The options on the **Imposition Options** page allow for the setting of imposition repetition, order, margins and markings.

As Imposition selections are very specific and can be quite confusing, we have provided a handy dynamic diagram that displays a representation of the current Imposition selections in real time. Whenever selections are changed, this display changes to reflect the selections made.

You can even select which specific pages are to be represented (up to page 1,000), whilst page numbers appear on the pages in the diagram and change dynamically, like everything else.

The Imposition selections that can be made are as follows:

- **Sheet Size** group:
 - **Final Media Size:** Use the drop-down to select the size of the media where the output is printed. The size of the media should be equivalent to the initial Section size multiplied by the number of repetitions, added with the margins and spaces between the repetitions.

If *Custom* media size is selected, enter the custom **Width** and **Height** values.

Note

The Sheet Size cannot be altered if a Cover Page was selected in the "Booklet Options" on page 857 Page.

- **Orientation:** Select orientation (aspect ratio) of media (Landscape or Portrait), or allow Connect to automatically determine the proper aspect ratio (Auto-Rotate).
- **Position:** Select from following options:

Note

If "Booklet Options" on page 857 were selected, then the **Position** settings are pre-set and cannot be altered here.

- **Auto-positioned:** This option creates unscaled imposition-ed pages.
- **Scale to fit:** Scales the imposition-ed pages so they fit on the N-Up stock. The scaled pages are then auto-positioned as usual.
- **Offset:** Allows for the selection of an offset position. The imposition-ed pages will be laid out so that the top left corner of the top left imposition-ed page is located at the selected offset.
If Offset is chosen, then the **Left Offset** and **Top Offset** selection boxes become active.

Note

The offset measures from the top left of the physical N-Up sheet to the top left imposition-ed page. If Auto-Rotate is selected (causing the N-Up stock to be rotated to fit the imposition-ed pages) then the measurement becomes the top left position of the rotated stock. i.e. The top left corner does not rotate with the stock.

- **Rotate final output Sheet 180 degrees (upside down):** Select to flip the output upside down.
- **Repetition group:**
Allows selection of how many Sections are to be placed, both Horizontally and Vertically. This is the total number of items, not the number of additional items being placed.

Note

If *Booklet Binding* were selected, some of these settings will be determined by the options made within the "Booklet Options" on page 857 Page and they cannot be altered here.

- **Gap group:**
Allows selection of the amount of blank space (either **Horizontal** and/or **Vertical**) to add between each page.

Note

If *Booklet Binding* were selected, some of these settings will be determined by the options made within the "Booklet Options" on page 857 Page and they cannot be altered here.

- **Order group:**

Note

If *Booklet Binding* were selected, some of these settings will be determined by the options made within the "Booklet Options" on page 857 Page and they cannot be altered here.

- **Page Order:** Select in which direction to go when adding sections to the output:
 - **Left to right, then top to bottom**
 - **Right to left, then top to bottom**
 - **Top to bottom, then left to right**
 - **Top to bottom, then right to left**
- **Stack Depth:** Stack Depth defines the number of sheets that will be printed before drill sorting resets.

Stack Depth works in conjunction with the Repetition, Gap and Page Order selections in order to determine the printing order

It is best illustrated by way of example.

The following example has a job with 100 single page documents and Repetition settings of 5 Horizontal and 2 Vertical, plus a Stack Depth of 5. Each table represents a single sheet, and the red line between the tables represents the Stack Depth:

1	6	11	16	21
26	31	36	41	46
2	7	12	17	22
27	32	37	42	47
3	8	13	18	23
28	33	38	43	48
4	9	14	19	24
29	34	39	44	49
5	10	15	20	25
30	35	40	45	50
51	56	61	66	71
76	81	86	91	96
52	57	62	67	72
77	82	87	92	97

Each set of 5 sheets is a discrete drill sorted nUp block that can be guillotined and stacked. The first 5 sheets encompass records 1-50, whilst the next 5 sheets encompass records 51-100.

The following example is the same job but with a Horizontal Repetition setting of 6:

1	6	11	16	21	26
31	36	41	46	51	56
2	7	12	17	22	27
32	37	42	47	52	57
3	8	13	18	23	28
33	38	43	48	53	58
4	9	14	19	24	29
34	39	44	49	54	59
5	10	15	20	25	30
35	40	45	50	55	60
61	66	71	76	81	86
91	96				
62	67	72	77	82	87
92	97				

As you can see, the outputs are *very* different.

Note

If the Stack Depth is greater than the number of sheets in the entire job then you will get a single drill sorted block that can be guillotined and stacked to give the complete job in record order.

- **Reverse Pages:** Select this option to reverse the order of pages. This would print the final record on the first page and the first record on the last page.
- **Force simplex:** Select this option to make the output Simplex, rather than the Imposition default of Duplex.

Note

If *Booklet Binding* were selected, some of these settings will be determined by the options made within the "Booklet Options" on page 857 Page and they cannot be altered here.

- **CropMarks** group:
 - **Type:** Use the drop-down to select the type of Crop Marks to add to the page.
 - **Page side:** What side(s) of the page to put the Crop Marks.
 - **Width:** Select the width of the crop mark lines.
 - **Offset:** How much separation (if any) to leave between the vertical and horizontal corner markings.
 - **Length:** Select the Length of the crop mark lines.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.

- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.
- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.



Insertor Options

The **Insertor Options** page allows the selection of a High Capacity Feeder (HCF) model. These machines are also commonly referred to as Inserters or Folder-Inserters.

The options available on this page are dependent upon the model selected.

The options selected on this page influence the position of the markings set on the next page: **"Mark Position Options" on page 867.**

- **Model:** Use the drop-down to select from any previously loaded Inserter model, or use the Browse button to select a HCF file to load a new Inserter model.
An image representing the chosen folder-inserter is displayed under the list, along with the HCF file details.
- **Options Group:**
The options available here are all Inserter dependent, and thus will change based upon the Inserter model selection.
To see how the selected Inserter markings would look on the printed page, click the Next button to move to the "Mark Position Options" on page 867 page, which has a preview of the page. You can move back and forward between these two pages until you are entirely satisfied with the selections made.
 - **Mark Configuration:** Use the drop-down to select the type of markings to add. This selection basically equates to the amount of area the markings will take up on the printed page.
 - **Fold Type:** Use the drop-down to select the type of fold to apply to the paper. This will impact upon where on the page the markings will be placed.
 - **Collation level:** Select whether the markings will be made at Document level, or Document Set level.
 - **Print marks on back:** Check to place the Inserter Marks on the rear of the page.

- **Selective Inserts:** If selective inserts are supported by the chosen Mark Configuration you can select what markings to include and whether those markings are to included based upon some conditional setting. This can be done by highlighting the Mark Name entry and either pressing the  **Edit** button, or using the right mouse click context menu, and selecting  **Edit**.
For information on how to edit the Selective Inserts settings, see Selective Insert Dialog
- **Clear Background Area Tab:**
Check the **Clear Background Area** checkbox to add a white background to the OMR, preventing background colors or elements interfering with the OMR Markings when they are read by the Inserter.
 - **Margins:**
 - **Same for all sides:** Check so that the Left margin selection is used to set all sides identically.
 - **Left, top, right, bottom:** Enter measurements for the margins on each side of the OMR Marks.
- **Custom OMR mark sizing Tab:**
If supported by the currently chosen Mark Configuration you can select a Custom OMR size by checking the **Custom OMR mark sizing** checkbox.
Select from any of the following, or leave the entries blank to use default values:
 - **Line length:** Enter a value between 10.16mm and 20mm.
 - **Line thickness:** Enter a value between 0.254mm and 0.63mm.
 - **Gap distance:** Enter a millimeter value 2.91mm and 4.2mm.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.

- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Mark Position Options







The **Mark Position Options** page displays a Preview of the output and the possible locations to place the inserter marks. The initial settings are determined by the selections made within the "Inserter Options" on page 865 page.

You can move back and forward between these two pages to perfect the settings, or you could move the inserter mark box to the desired location on the preview.

Preview box:

- The *pink area* displays the areas of the page where inserter marks can be positioned.
- The *small checkered box* displays the current location of the inserter marks. This box is selectable and can be dragged to the desired location within the printable (pink) areas. If the box is placed outside the printable areas the page will display an error and prevent attempts at leaving the page.

Below the Preview box are buttons which allow control of the Preview box. The selections that can be made are:

-  **First Page**: Click to jump to the first page.
-  **Previous Page**: Click to move to the previous page.
-  **Next Page**: Click to move to the next page.
-  **Last Page**: Click to jump to the last page.
- **Show Page**: Use the up and down arrows or type a page number to display a specific page within the document.
-  **Zoom in/out**: Click to zoom in or out by 25%
-  **Zoom Level**: Use the drop-down to select a predefined level or enter a zooming percentage.

PDF Options

The **PDF Options** page is shown only when a PDF Print output type is selected in the [Print Options](#) dialog. It is used to select PDF specific options.

- **PDF Options Group**

- **PDF Type:** Use the drop-down to specify which format the PDF should be generated in. These options are standard PDF, archive format PDF (PDF/A-1b), graphics format PDF (PDF-X4) and variable data printing format PDF (PDF-VT).
- **Embed standard fonts:** Click to embed the 14 standard system fonts within the PDF output. This increases the output filesize but makes the PDF output truly portable. Such PDFs print as displayed on screen, regardless of whether the 14 standard fonts are present on the target printing system or not.

Note

This box is ignored for PDF/A and PDF-X4 output, as fonts are always embedded in those output types.

- **Pass-through PDF resources:** Click to have PDF resources used *as-is*, without any additional processing. This guarantees the fidelity of any PDF graphics used within the template will be retained in the output.

Note

Connect tries to write content in the best way possible, depending on the chosen output format and optimization settings. Selecting *PDF pass-through* means the output will be less optimized, which typically produces somewhat larger files.

- **Add Digital Signature Group:** Check to enable the integration of a digital signature into the PDF.













A digital signature identifies the person signing a document, similarly to a conventional handwritten signature. Unlike a handwritten signature, a digital signature is difficult to forge as it contains encrypted information which is unique to the signer and which can be password protected and verifiable.

- **All Keystores:**

Here you can choose from existing digital signatures, or select new ones.

- **Name:** The user-defined name of the keystore.
- **File:** The file path and name to the keystore file.

This is where you select keystore values.

-  **New:** Click to open the [Key Store](#) dialog to add a new keystore to the list.
 -  **Duplicate:** Click to make a copy of the currently selected keystore.
 -  **Edit:** Click to edit the currently selected keystore in the [Key Store](#) dialog.
 -  **Delete:** Click to delete the currently selected keystore.
 -  **Move Up:** Click to move the currently selected keystore up.
 -  **Move Down:** Click to move the currently selected keystore down.
- **All Signatures:** Displays a list of signatures to add to the PDF output.
 - **Name:** The user-defined name of the signature.
 - **File:** The file path and name to the signature file.
 - **Alias:** The user-defined alias for the signature.
 -  **New:** Click to open the [PDF Signature](#) dialog to add a new signature to the list.
 -  **Duplicate:** Click to make a copy of the currently selected signature.
 -  **Edit:** Click to edit the currently selected signature in the [PDF Signature](#) dialog.
 -  **Delete:** Click to delete the currently selected signature.
 -  **Move Up:** Click to move the currently selected signature up.
 -  **Move Down:** Click to move the currently selected signature down.

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.

- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.
- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

PDF Digital Signature Options

PDF Signature

The **PDF Signature** dialog appears when adding or editing a signature from the "PDF Options" on page 868 page.

- **Name**: Enter a name that describes the signature entry.
- **Keystore**: Use the drop-down to select which keystore the signature is pulled from. These keystores are set in the "Keystore" on page 872 dialog, called from the "PDF Options" on page 868 page.
- **Signature Properties group**: These are optional Metadata fields associated with the signature, which can be omitted.
 - **Location**: The CPU host name or physical location of the signing.
 - **Reason**: Records the reason for the signing.
 - **Contact**: Information to enable a recipient to contact the signer to verify the signature. For example: a phone number.
 - **Handler**: The PDF reader plugin used to interpret the signature data. It should be left at its default setting (Adobe.PPKLite) unless time-stamping is desired, in which case "Adobe.PPKMS" is likely the best option.
- **Key group**: Refers to a key from the keystore.
 - **Alias**: The user-friendly name of the key
 - **Password**: Enter the password for the key (the same password as was entered in [Key Store](#)).
 - **Repeat Password**: Re-enter the password for the key (same as previous).

- **Apply Time Stamping Authentication group:** Check to enable time stamping authentication.

Note

Not available for signatures set to use Adobe.PPKLite Handler.

- **URL:** Select the Time Stamp Authority (TSA) URL address.
- **Account:** Account name specific to the TSA server chosen.
- **Password:** Password specific to the TSA server chosen.
- **Repeat Password:** Repeat of password.
- **Visible Signature group:** Check to add a visible signature to the PDF file.
 - **X:** Enter the horizontal distance between the left side of the page and the left side of the signature, in points (pt).
 - **Y:** Enter the vertical distance between the top of the page and the top of the signature, in points (pt).
 - **Width:** Enter the desired width of the signature, in points (pt).
 - **Height:** Enter the desired height of the signature, in points (pt).

Advanced Print Wizard navigation options

- **Load** button: Click to select a previously created Output Creation Preset. This will change the Advanced Print Options to match the entries contained within the Preset.
- **Preview** button: Click to launch a [Proof Preview](#) window, which displays how the printed output would look based upon the currently chosen selections.
- **Back** and **Next** buttons: Used to navigate back and forth through all the selected options within the Wizard. Up until the Print button is pressed, one can reverse all the way through the wizard to return to the main selection page (the "Print Options" on page 851 page) and add or remove printing options from the print run.
- **Print** button: Click to produce print output according to the current settings. This can be done at any point within the Wizard, whether or not the options selected in the the "Print Options" on page 851 page have been completed or not.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

Keystore

The security certificate **Keystore** dialog appears when adding or editing a keystore from the "PDF Options" on page 868 page.

This dialog allows you to select a keystore with a private key.
The keystores currently supported by Connect are:

- JKS (Java Key Store) format.
- PKCS#12
- PKCS#11

Note

PKCS#11 requires an extra plug-in not included in the PlanetPress Connect installation.

These are the options available in this dialog:

- **Name:** Enter a name for the keystore to describe it within Connect.
- **File:** Enter the path to the keystore file, or use the Browse button to locate the file.
- **Keystore properties group:**
 - **Type:** Use the drop-down to select the appropriate type of the keystore format the file is: JKS, PKCS11, PKCS12.
 - **Provider:** Enter the provider of the keystore.
 - "SUN" for JKS
 - "SunJSSE" for PKCS#12
 - "IAIK PKCS#11:1" for PKCS#11
 - **Password:** Type in the password that secures the keystore, if the keystore is password protected.
 - **Repeat Password:** Re-type in the password that secures the keystore. Once this is done the two Password entry boxes will no longer have the red cross icon (indicating incomplete or unselected) flag beside them.
- **Properties file group:**

- **File:** Load optional keystore properties file. Could be used to store the password in a file.

Designer Script API

In Designer templates, every bit of information can be tailor-made, using scripts. When Connect generates actual output – letters, web pages or emails -, it opens a record set and merges it with the template. It takes each record, one by one, and runs all scripts for it (in a specific order, see "The script flow: when scripts run" on page 660).

Most scripts can be made using one of the Script Wizards (see "Personalizing Content" on page 592).

However, when you want to do more than what you can do with a Wizard, you may write a script yourself. If you are not familiar with writing scripts, please read "Writing your own scripts" on page 624 first.

All scripts in the Designer have to be written in JavaScript.

If you don't know JavaScript, the many examples given in this API will help you get started.

It is worth the effort, however, to familiarize yourself with the JavaScript syntax. For a simple script all you need to know can be found on the following web pages:

http://www.w3schools.com/js/js_syntax.asp and http://www.w3schools.com/js/js_if_else.asp.

In the editor window, press **Ctrl + Space** to see the available features and their descriptions.

Use the arrow keys to select a function or object and press Enter to insert it in the script.

Type a **dot** after the name of the function or object and press Ctrl + space again to see which features are subsequently available.

For more keyboard shortcuts, see "Keyboard shortcuts" on page 738.

Designer API

The "Designer Script API" on the facing page describes the objects and functions that are available in **template scripts**, created inside the Scripts pane. Template scripts change the contents of sections in a template.

Note

In a Print context, the scripts in the Scripts pane run once for each section and once for each Master Page (see "Master Pages" on page 350).

Control Script API

Control Scripts are a special kind of Designer Scripts. They don't touch the content of the sections themselves, but they change the way a template is outputted, for example by selecting or omitting sections from the output.

For more information about Control Scripts and their use, see "Control Scripts" on page 645. Features that are specific to Control Scripts are listed in the "Control Script API" on page 930.

Designer Script API

This page lists the global objects and functions that are available in scripts, created inside the Scripts pane. Click through to an object or function to get a description and examples.

If you are not familiar with writing scripts, see "Writing your own scripts" on page 624.

Control Scripts

Control Scripts are a special kind of Designer Scripts. They don't touch the content of the sections themselves, but they change the way a template is outputted, for example by selecting or omitting sections from the output. For more information about Control Scripts and their use, see "Control Scripts" on page 645. Features that are specific to Control Scripts are listed in the "Control Script API" on page 930.

Objects

Object	Description
"results" on page 878	<p>This object is used to manipulate the content of the template. It contains the HTML element or set of HTML elements that match the selector of the script, specified in the script editor.</p> <p>This object is not available in Control Scripts, because that type of script doesn't have a selector (see "Control Scripts" on page 645).</p>

Object	Description
"record" on the facing page	The record in the main data set that is currently being merged. To get the value of a field in the record, use <code>record.fields['fieldname']</code> or <code>record.fields.fieldname</code> .
"logger" on the facing page	Global object that allows you to log messages.
locale	Defines which locale to use. See "Locale" on page 590.
"formatter" on page 912	Global object that allows you to format values (such as a date or number).
"automation" on page 931	This object encapsulates the properties of the Workflow process that triggered the current operation. Not available in PrintShopMail Connect.
"merge" on page 935	The <code>merge</code> object is mainly used in Control Scripts. It gives access to the template with all of its contexts and sections. It doesn't give access to the content of the sections. To change the content of a section, you would create a script with a selector and use the <code>results</code> object in the script (see "results" on page 878).

Global functions

Function	Description
"loadhtml()" on page 920	Loads HTML data from a HTML (snippet). The returned HTML can be placed into a variable or into a set of HTML elements.
"loadjson()" on page 922	Loads json data from a URL. This is a simple way to retrieve content from external systems.
"query()" on page 926	Performs a query in the template's contents and creates a new result set containing the HTML elements that match the given CSS selector.

Function	Description
"resource()" on page 928	This function returns information about an image resource. It can also be used to check if a file exists.

Examples of iterator functions

Function	Description
"Each" on page 909	A generic iterator function, to iterate over the elements in the result set
"For...in" on page 911	Iterates over the enumerable properties of an object, in arbitrary order. For each distinct property, statements can be executed.

logger

This is a global `ScriptLogger` object that allows logging messages such as error, warning or informational messages. The messages will appear in the **Messages** pane (see "Preflight Results and Messages" on page 759 and "Designer User Interface" on page 666).

Methods

These are the methods of the logger object.

Method	Parameters	Description
error()	message: string	Logs an error message
info()	message: string	Logs an informational message
warn()	message: string	Logs a warning message

record

The `record` object gives access to the record that is currently being merged with the template.

Properties

Field	Type	Description
fields	Array	The field values that belong to this record. You can access a specific field value using either a numeric index or the field name: <code>record.fields['fieldname']</code> or <code>record.fields.fieldname</code> .
id	Number	The id of this record.
index	Number	The one-based index of this record, or zero if no data is available.
tables	Array	The detail tables that belong to this record. You can access a specific table using either a numeric index or the table name, followed by a numeric index for a record inside that detail table. For example, to access the value of the field <code>prod_id</code> in the first record of a detail table called <code>detail</code> , use: <code>record.tables["detail"][0].fields["prod_id"]</code> .

Examples

The following template script evaluates the data field `Country` in the current `record`. If the value is 'CANADA' it will show the `results`, otherwise it will hide them. (The `results` object contains the elements that match the script's selector; see "results" on the facing page and "Writing your own scripts" on page 624.)

```
if (record.fields["Country"] == "CANADA") {
    results.show();
} else {
    results.hide();
}
```

In a Control Script, an entire section could be enabled or disabled based on the same condition:

```
if (record.fields["Country"] == "CANADA") {
    merge.template.contexts.PRINT.sections["Section 1"].enabled =
true;
} else {
```

```
merge.template.contexts.PRINT.sections["Section 1"].enabled =
false;
}
```

(For more information about Control Scripts, see "Control Scripts" on page 645.)

The next script looks up a value in the first record in a detail table and shows or hides the results depending on that value.

```
if (record.tables["detail"][0].fields["prod_id"] == "10") {
    results.show;
} else {
    results.hide;
}
```

Note that indexes start counting at 0, so `tables["detail"][0]` refers to the first record in the detail table.

results

The `results` object (type: `QueryResults`) is the result of the query for HTML elements that match the selector of the script. The selector of a script can be specified in the Script Editor and is visible in the second column of the Scripts pane, next to the name of the script.

If, for example, a script would have the selector `p.onlyCanada`, the script would apply to all paragraphs that have the class `onlyCanada`. (Classes can be defined in the **Attributes** pane at the right: select the element in the content and type the class(es) in the **Class** field.)

The script could then use the `results` object to hide or show those paragraphs, depending on the value of the data field `Country` in the current record:

```
if (record.fields["Country"] == "CANADA") {
results.show();
} else {
results.hide();
}
```

Note

This object can't be used in Control Scripts, because they don't have a selector.

Property

Field	Type	Description
length	Number	Number of elements in this result set. Equivalent to calling size().

Functions

The functions below can be called by the `results` object and by any other result set that is returned by a query, see "query()" on page 926.

Function	Description
"Examples" on page 882	Adds elements to a set of HTML elements.
"Examples" on page 883	Adds the specified class to each element in a set of HTML elements. Has no effect if the class is already present.
"Examples" on page 884	Inserts content after each element in a set of HTML elements..
"Examples" on page 886	Inserts content at the end of each element in a set of HTML elements.
"Examples" on page 889	Change the given attribute of the element or set of HTML elements with the given value.
" Examples" on page 890	Inserts content before an element or before each element in a set of HTML elements.
"Examples" on page 892	Returns the immediate children of an HTML element.
"Examples" on page 944	Returns a new result set containing a copy of each element in a set of HTML elements.

Function	Description
"Examples" on page 894	For each element in a set, this function gets the first parent element that matches a selector, by testing the element itself and traversing up through its ancestors in the DOM tree.
"Examples" on page 895	Gets the value of a style property for the first element in set of HTML elements or sets one or more CSS properties for every element in a set of HTML elements.
"Example" on page 897	Removes the contents (child elements and inner HTML) from one element or a set of elements in the template.
"Example" on page 897	Returns a subset of the current result set.
"Example" on page 898	Performs a search for a text in the children of each element in a set of HTML elements, and returns a new result set with elements that surround the occurrences.
"Example" on page 899	Returns <code>true</code> if the first element in this result set has the specified class.
"hide()" on page 899	Hides the HTML element or set of HTML elements.
"Examples" on page 900	Replaces the inner HTML of the element or of each element in a set of HTML elements with the supplied value, or returns the HTML of the first element if no value is supplied.
is(selector)	Returns true if at least one of the elements in a set of HTML elements matches the supplied CSS selector.
"Creating a table of contents" on page 900	Returns a marker that will be replaced with the element's page number after pagination. This only works for elements in the section that is currently being merged.
"Example" on	Returns the parents of the elements in a set of HTML elements.

Function	Description
page 902	
"Examples" on page 902	Inserts content at the beginning of an HTML element or of each element in a set of HTML elements.
"Examples" on page 905	Removes an HTML element or a set of HTML elements from the document.
"Examples" on page 906	Removes the specified attribute from each element in this result set.
"Examples" on page 906	Removes the specified class from an element or from each element in a set of HTML elements. Has no effect if the class is not present.
"Replace elements with a snippet" on page 907	Replaces an HTML element or a set of HTML elements (with a snippet, for example). Returns the result set.
"show()" on page 907	Shows the HTML element or a set of HTML elements.
size()	Gets the number of elements in this result set. Equivalent to the <code>length</code> property.
"Example" on page 908	Replaces the text content of an HTML element or of each element in a set of HTML elements with the supplied value, or returns the text content of the first element if no value is supplied.

add()

The `add()` function allows you to add elements to a set of HTML elements that match the selector of the script or of another query in the template (see "`query()`" on page 926).

add(content)

Returns the union of this result or result set and other content.

content

A query result. This can be an HTML string or a result set.

Examples

Add one result set to another

This script adds one query result to another and sets the background color to yellow.

```
query("#test1").add(query("#test2")).css("background", "yellow");
```

Note: the way the functions `add()` and `css()` are used in this script is called 'chaining'. Chaining is optional; the same could be achieved by storing the results of the queries in a variable:

```
var myResult = query("#test1");
myResult.add(query("#test2"));
myResult.css("background", "yellow");
```

Creating an empty result set and adding elements to it

The following script loads snippets in an iteration and adds their elements to an empty result set (using `query()`). Then it replaces a placeholder in the template with the new result.

```
var chapters = query();
for ( var i = 1; i <= 4; i++) {
chapters = chapters.add(loadhtml('snippets/Chapter' + i +
'.html'));
}
results.replaceWith(chapters);
```

Selector	Matched element	Matched element after script execution
#chapters	<p id="chapters">{{chapters}}</p>	<h1>Chapter 1</h1> <p>Lorem ipsum...</p> <h1>Chapter 2</h1> <p>Lorem ipsum...</p> <h1>Chapter 3</h1> <p>Lorem ipsum...</p> <h1>Chapter 4</h1> <p>Lorem ipsum...</p>

addClass()

Adds the specified class(es) to each element in a set of HTML elements that match the selector of the script or of another query in the template (see "query()" on page 926). This has no effect if the class is already present.

addClass(classname)

Adds the specified class(es) to each element in a result set. Has no effect if the class is already present.

classname

String, space separated list of class names.

Examples

This script adds a class name to a paragraph.

```
results.addClass("foo");
```

Selector	Matched element	Matched element after script execution
p	<p>Hello world</p>	<p class="foo bar">Hello world</p>

The following script adds two class names to a paragraph.

```
results.addClass("foo bar");
```

Selector	Matched element	Matched element after script execution
p	<p>Hello world</p>	<p class="foo bar">Hello world</p>

after()

Insert content after each element in the set of HTML elements that match the selector of the script or of another query in the template (see "query()" on page 926). See also: "Examples" on page 890.

after(content)

Insert content after each element in the set of HTML elements that match the selector of the script, or of another query in the template (see "query()" on page 926). After creates a new result set.

content

String, HTML string or result set to insert after the matched elements. In case a plain text string is provided, it is automatically wrapped in a element to avoid orphan text nodes to appear in the <body> element.

Examples

This script looks up an element with the ID #salesrep and inserts a paragraph after it.

```
query("#salesrep").after("<p>Lorem ipsum</p>");
```

Matched element	Matched element after script execution
<code><p id="salesrep">Peter Parker</p></code>	<code><p id="salesrep">Peter Parker</p> <p>Lorem ipsum</p></code>

This script looks up an element with the ID #salesrep, sets its text color to red and inserts a paragraph after it.

```
query("#salesrep").after("<p>Lorem ipsum</p>").css("color","red");
```

Matched element	Matched element after script execution
<code><p id="salesrep">Peter Parker</p></code>	<code><p id="salesrep" style="color: red;">Peter Parker</p> <p>Lorem ipsum</p></code>

Note: the way the functions after() and css() are used in this script is called 'chaining'. Chaining is optional; the same could be achieved by storing the result of the query in a variable:

```
var salesrep = query("#salesrep");  
salesrep.after("<p>Lorem ipsum</p>");  
salesrep.css("color","red");
```


The following script inserts a paragraph after the elements in the `results` (the set of HTML elements that match the selector of the script).

```
results.after("<p>Lorem Ipsum</p>");
```

Matched element	Matched element after script execution
<code><p id="salesrep">Peter Parker</p></code>	<code><p id="salesrep">Peter Parker</p></code> <code><p>Lorem ipsum</p></code>

This script looks for the string "Lorem " in the `results` (the set of HTML elements that match the selector of the script) and inserts the string "ipsum" right after that text. The string is automatically enclosed in a span.

```
results.find("Lorem ").after("ipsum");
```

Matched element	Matched element after script execution
<code><p>Lorem dolor sit amet, consectetur adipiscing elit.</p></code>	<code><p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p></code>

This script looks up an element with the ID `#salesrep` and inserts a string after it. The string is automatically enclosed in a span.

```
query("#salesrep").after("Lorem Ipsum");
```

Matched element	Matched element after script execution
<code><p id="salesrep">Peter Parker</p></code>	<code><p id="salesrep">Peter Parker</p></code> <code>Lorem Ipsum</code>

append()

Insert content at the end of each element in the set of each element in a set of HTML elements that match the selector of the script or of another query in the template (see "query()" on

page 926). See also: "Examples" on page 902.

append(content)

Insert content as the last element to each element in the set of HTML elements that match the selector of the script or of another query in the template (see "query()" on page 926). `Append` creates a new result set.

content

String, HTML string or result set to insert after the elements. In case a plain text string is provided, it is automatically wrapped in a `` element to avoid orphan text nodes to appear in the `<body>` element.

Examples

This script appends a paragraph to the `results` (the set of HTML elements that match the selector of the script).

```
results.append("<p>Peter Parker</p>");
```

Selector	Matched element	Matched element after script execution
<code>#box</code>	<pre><div id="box"> <h1>Personal information</h1> </div></pre>	<pre><div id="box"> <h1>Personal information</h1> <p>Peter Parker</p> </div></pre>

This script appends a string to the `results` (the HTML elements that match the selector of the script). The string is added to the end of the matched element(s) and wrapped in a `Span` element.

```
results.append("Peter Parker");
```

Selector	Matched element	Matched element after script execution
<code>.name</code>	<pre><div> <h1>Personal information</h1> <p></pre>	<pre><div> <h1>Personal information</h1> <p class="name">Name: Peter Parker</p></pre>

Selector	Matched element	Matched element after script execution
	<pre>class="name">Name: </p> </div></pre>	<pre></div></pre>

This script's selector is `<div>`, so the script appends a paragraph to all Div elements in the template.

```
results.append("<p>Peter Parker</p>");
```

Selector	Matched element	Matched element after script execution
<code>div</code>	<pre><div> <h1>Personal information</h1> </div> <div> <h1>Personal information</h1> </div></pre>	<pre><div> <h1>Personal information</h1> <p>Peter Parker</p> </div> <div> <h1>Personal information</h1> <p>Peter Parker</p> </div></pre>

The following script appends a snippet to a Div element with the ID `box`.

```
var a = loadhtml('snippets/snippet_name.html');
results.append(a);
```

Selector	Matched element	Matched element after script execution
<code>#box</code>	<pre><div id="box"> <h1>Personal information</h1> </div></pre>	<pre><div id="box"> <h1>Personal information</h1> <p>Peter Parker</p> </div></pre>

This script looks for an element with the ID `box` and appends a paragraph to it.

```
query("#box").append("<p>Peter Parker</p>");
```

Matched element	Matched element after script execution
<pre><div id="box"> <h1>Personal information</h1> </div></pre>	<pre><div id="box"> <h1>Personal information</h1> <p>Peter Parker</p> </div></pre>

This script looks for an element with the ID `box`, appends a paragraph to it and colors all text inside the box red.

```
query("#box").append("<p>Peter Parker</p>").css("color", "red");
```

Matched element	Matched element after script execution
<pre><div id="box"> <h1>Personal information</h1> </div></pre>	<pre><div id="box" style="color: red;"> <h1>Personal information</h1> <p>Peter Parker</p> </div></pre>

Note: the way the functions `append()` and `css()` are used in this script is called 'chaining'. Chaining is optional; the same could be achieved by storing the result of the query in a variable:

```
var box = query("#box");
box.append("<p>Peter Parker</p>");
box.css("color", "red");
```

attr()

Returns the value of the specified attribute of the first element in a result set, or sets the value of the specified attribute of each element in a result set.

`attr(attributeName): String`

Returns the value of the specified attribute of the first element in a result set.

attributeName

String; the name of the attribute.

Examples

This script - with the selector `img` - stores the source of the first image in a variable.

```
var src = results.attr("src");
```

The following script looks up an image with the ID `#image1` and stores its background color in a variable.

```
var imgURL = query("#image1").attr("src");
```

`attr(attributeName, value)`

Sets the value of the specified attribute of each element in a result set.

attributeName

String; the name of the attribute.

value

String; value for the attribute.

Examples

This script looks up an image in an element with the ID `#calloutbox` and sets its alternative text to a value from a data field

```
var altText = record.fields.FavHobby;
query("#callout img").attr('alt', altText);
```

The following script sets the background color of a specific table cell in an email to red if the value of the field `TOTAL` has a negative value in the current record.

```
if(record.fields.TOTAL<0) {
    query("#total").attr("bgcolor", "red");
}
```

before()

Insert content before each element in the set of HTML elements that match the selector of the script or of another query in the template (see `query()` on page 926). See also: "Examples" on page 884.

before(content)

Before(content) inserts content before each element in the set of elements that match the script's selector. Before() creates a new result set.

content

String, HTML string or result set to insert after the elements. In case a plain text string is provided, it is automatically wrapped in a element to avoid orphan text nodes to appear in the <body> element.

Examples

This script looks for an element with the ID `salesrep` and inserts a paragraph before that element.

```
results.before("<p>Lorem Ipsum</p>");
```

Selector	Matched element	Matched element after script execution
<code>#salesrep</code>	<code><p id="salesrep">Peter Parker</p></code>	<code><p>Lorem ipsum</p></code> <code><p id="salesrep">Peter Parker</p></code>

This script does the same, but it uses the `query()` function to look up the element.

```
query("#salesrep").before("<p>Lorem ipsum</p>");
```

Matched element	Matched element after script execution
<code><p id="salesrep">Peter Parker</p></code>	<code><p>Lorem ipsum</p></code> <code><p id="salesrep">Peter Parker</p></code>

The following script looks for an element with the ID `salesrep`, inserts a paragraph before that element and colors that element red.

```
query("#salesrep").before("<p>Lorem ipsum</p>").css("color", "red");
```

Matched element	Matched element after script execution
<code><p id="salesrep">Peter Parker</p></code>	<pre> <p >Lorem ipsum</p> <p id="salesrep" style="color: red;">Peter Parker</p> </pre>

Note: the way the functions `before()` and `css()` are used in this script is called 'chaining'. Chaining is optional; the same could be achieved by storing the result of the query in a variable:

```

var salesrep = query("#salesrep");
salesrep.before("<p>Lorem ipsum</p>");
salesrep.css("color", "red");

```

The following script searches the results for the string "ipsum" and puts "Lorem " before it. "Lorem " is automatically wrapped in a Span element.

```

results.find("ipsum").before("Lorem ");

```

Matched element	Matched element after script execution
<code><p>ipsum dolor sit amet, consectetur adipiscing elit.</p></code>	<pre> <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p> </pre>

The following script looks for an element with the ID salesrep and inserts the text "Lorem Ipsum" before that element. "Lorem Ipsum" is automatically wrapped in a Span element.

```

query("#salesrep").before("Lorem Ipsum");

```

Matched element	Matched element after script execution
<code><p>ipsum dolor sit amet, consectetur adipiscing elit.</p></code>	<pre> Lorem Ipsum <p id="salesrep">Peter Parker</p> </pre>

children()

Returns the immediate children (inner HTML) of the elements in a result set.

Examples

This script retrieves the inner HTML of an element selected from a snippet.

```
var snippet = loadhtml('snippets/snippet.html', '#foobar').children();
results.append(snippet);
```

The following script retrieves the inner HTML of the elements and then performs a find/replace.

```
var snippet = loadhtml('snippets/snippet.html', '#foobar').children();
snippet.find('@firstname@').text('foobar');
results.append(snippet);
```

clone()

This function returns a new set containing a copy of each element in a set; see "Dynamically adding sections (cloning)" on page 655.

Note

Due to resource constraints, the number of unique clones that can be created throughout a job is limited to around 20. A clone is considered unique if it has a different name. This is a rough estimate; if the template is simple, up to 60 clones may be created.

The limit only applies to the amount of unique clones. There is no limit to the amount of `clone()` function calls.

To duplicate an existing template element, clone it before calling `append()`; see "Examples" on page 886.

Examples

This script performs an iteration over the elements in the `results` (the elements that match the selector of the script).

```
var row = query("tbody tr", results).clone();
query("tbody", results).append(row);
```


The following script clones an existing table row to match the number of rows in a detail table. Afterwards it iterates over the rows to populate the fields.

```
// Create the number of rows based on the records in the detail
table
// We start at 1 so the boilerplate row is used too and there is no
need to delete that row
for(var r = 1; r < record.tables['detail'].length; r++) {
results.parent().append(results.clone());
}

// Iterate over the rows and populate them with the data from the
accompanying data row
query("#table_2 > tbody > tr").each(function(i) {
this.find('@ItemNumber@').text( record.tables['detail'][i].fields
["ItemNumber"]);
this.find('@ItemOrdered@').text( record.tables['detail'][i].fields
["ItemOrdered"]);
this.find('@ItemTotal@').text( record.tables['detail'][i].fields
["ItemTotal"]);
this.find('@ItemDesc@').text( record.tables['detail'][i].fields
["ItemDesc"]);
this.find('@nr@').text(i);
});
```

The following script clones and populates a boilerplate row. Once completed you will need to hide the boilerplate row.

```
for(var i = 0; i < record.tables['detail'].length; i++) {

var row = results.clone(); //Clone our boilerplate row

row.find('@ItemNumber@').text( record.tables['detail'][i].fields
["ItemNumber"]);
row.find('@ItemOrdered@').text( record.tables['detail'][i].fields
["ItemOrdered"]);
row.find('@ItemTotal@').text( record.tables['detail'][i].fields
["ItemTotal"]);
row.find('@ItemDesc@').text( record.tables['detail'][i].fields
["ItemDesc"]);
row.find('@nr@').text( i );

results.parent().append(row);
```

```
}  
  
// Hide our boilerplate row (note that this doesn't really delete  
the row).  
results.hide();
```

closest()

For each element in a set, this function gets the first parent element that matches a selector, by testing the element itself and traversing up through its ancestors in the DOM tree. (In HTML, a parent is an element that contains another element.)

To get a child element or all child elements, use `children()` (see "Examples" on page 892).

The `closest()` command is based on the `closest()` command found in the jQuery library: <https://api.jquery.com/closest/>.

closest(selector)

For each element in a set, this function gets the first element that matches the selector by testing the element itself and traversing up through its ancestors in the DOM tree.

selector

A String containing an HTML tag (without the angle brackets, <>).

Examples

The following script looks up all table rows in the template that contain an `<input>` element.

```
query("input").closest("tr");
```

This code gets the closest 'parent' row for each element that matches the selector of the script (collected in the `results` object):

```
results.closest("tr");
```

The rows could be coloured red within the same statement:

```
results.closest("tr").css('background-color', 'red');
```

css()

Gets the value of a style property for the first element in the set of HTML elements that match the selector of the script or of another query in the template (see "query()" on page 926), or sets one or more CSS properties for every element in the set.

css(styleName) : String

Returns the value of the specified CSS property.

propertyName

String; the name of the CSS property.

Examples

This script stores the text color of the `results` (the HTML elements that match the selector of the script) in a variable.

```
var textcolor = results.css("color");
```

The following script looks up an element with the ID `#calloutbox` and stores its background color in a variable.

```
var backgroundcolor = query("#calloutbox").css("background-color");
```

css(styleName, value)

Function to set a CSS property.

propertyName

String; the name of the CSS property.

value

String; value for the CSS property or a map of property-value pairs to set.

Examples

This script looks up an element with the ID `#calloutbox` and sets its text color to red.

```
query("#callout p").css('color' , 'red');
```

The following script does the same, but it only sets the text color to red if in the current record the value of the field 'accounttype' is 'PRO'.

```
if(record.fields.accounttype == "PRO") {  
  query("#callout p").css("color","red");  
}
```

This script sets the text color of the results to a hexadecimal color code.

```
results.css('color' , '#669900');
```

This script loads a snippet into a variable. Then it finds/replaces text in the snippet and applies a CSS property to the replacing text.

```
var mysnippet = loadhtml('snippets/snippet vars.html');  
mysnippet.find('@var@').text('OL Connect').css('text-decoration','underline');  
results.replaceWith(mysnippet);
```

css(properties)

Function to set one or multiple CSS properties.

properties

Array; map of property-value pairs to set.

Examples

This script colors the text of the `results` (the set of HTML elements that match the selector of the script) red and makes it bold.

```
results.css({'color' : 'red', 'font-weight' : 'bold'});
```

empty()

Remove the contents (child elements and inner HTML) from one element or a set of elements in the template.

Use `remove()` to remove the elements themselves.

Example

This script empties all Span elements found in the template.

```
results.empty();
```

Selector	Paragraph before script execution	Paragraph after script execution
span	<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit.</p>	<p>Lorem ipsum amet, consectetuer adipiscing elit.</p>

filter()

filter(callback)

Returns a subset of a set. All elements for which the callback function returns `true` will be included in the result.

callback

A function used as a test for each element in the set. Filter() passes the iteration index and the current element to the callback function. In the scope of the callback function, `this` refers to the current element.

Example

The selector of the following script is `li` (list item), so the `results` object contains all list items in the template. The script filters the third and sixth line items from the `results`, taking advantage of the index that is passed to the filter function, and colors them red. It uses the modulus operator (%) to select every item with an index value that, when divided by 3, has a remainder of 2. (The index starts counting at zero.)

```
results.filter(function(index) {
    return index % 3 === 2;
}).css( "background-color", "red" );
```

filter(selector)

Returns a subset of a set. All elements matching the selector will be included in the result.

The difference between `results.filter(selector)` and `query(selector, results)` is that `query()` searches throughout the entire `results` while `filter()` only takes the top-level elements into account.

selector

A String containing a CSS selector. See http://www.w3schools.com/cssref/css_selectors.asp for CSS selectors and combinations of CSS selectors.

Example

The selector of the following script is `tr` (table row), so the object `results` contains all rows in the template. The script filters all even rows from the `results` and colors them red.

```
results.filter(":nth-child(even)").css("background-color", "red");
```

find()

find(textToFind)

Performs a deep search for `textToFind` in the children of each element, and returns a new result set with elements that surround the occurrences.

textToFind

A String that contains the search text.

Example

The following piece of code loads a snippet, then looks for placeholders using `find()`, and replaces them with a text.

```
var mysnippet = loadhtml('snippets/snippet.html');
mysnippet.find('@var1@').text('OL Connect 1');
mysnippet.find('@var2@').html('<i>OL Connect 2</i>').css('text-
```

```
decoration', 'underline');
results.replaceWith(mysnippet);
```

hasClass()

hasClass(classname) : Boolean

Returns true if the first element in this result set has the specified class.

classname

String containing one class name.

Example

This script checks if the first of the `results` (the set of elements matching the selector of the script) has the class 'green'. If so, it colors the text of all the elements in the `results` green.

```
if (results.hasClass('green')) {
    results.css('color', 'green');
}
```

hide()

Hides the elements in a set. This doesn't remove the elements; to make them visible again, use the function "show()" on page 907.

These functions are used by the Conditional Script Wizard, as you can see when you open a Conditional Script and click the **Expand** button; see "Showing content conditionally" on page 613.

Example

This script hides or shows the elements matched by the selector of the script (which are stored in the `results` object), depending on the value of the data field `Country` in the current record.

```
if (record.fields["Country"] == "CANADA") {
results.show();
} else {
results.hide();
}
```

html()

html() : String

Returns the inner HTML of the first element in this result set.

html(value)

Replaces the inner HTML of each element in this result set by the supplied value.

value

A String that may contain HTML tags.

Examples

The following script loads part of a snippet based on the value of a field, and then inserts the content into the document using `html()`.

```
var promoTxt = loadhtml('snippets/promo-en.html', '#' +
record.fields['YOGA']);
results.html(promoTxt);
```

The following script loads a snippet. Then it looks for a placeholder (`@var2@`) in the text of that snippet and replaces every found placeholder by the text '`<i>OL Connect 1</i>`'. It uses `html()` so the HTML formatting (`<i>` and `</i>`) will indeed be interpreted as HTML. Finally, it places the snippet in the template.

```
var mysnipppet = loadhtml('snippets/snippet.html');
mysnipppet.find('@var1@').html('<i>OL Connect 1</i>');
results.replaceWith(mysnipppet);
```

pageRef()

Returns a **marker** that will be replaced with the element's page number after pagination. This only works for elements in the section that is currently being merged.

Example

Creating a table of contents

The following script creates a table of contents for all level 1 headings (`<h1>` elements) with the class `title` in one section.


```

var toc = '<ul ID="toc">';
query('h1.title').each(function()
{toc += '<li>' + this.text() + ' <span class="li_toc">' +
this.pagerref() + '</span></li>';
});
toc += '</ul>';
results.after(toc);

```

The first line creates a variable for the table of contents, which will be a list (a `` element with the ID `toc`). The start tag of the list is added to the variable.

The next line does a query for all level 1 headings (`<h1>` elements) with the class `title` in the current section. With `each()` the script loops through them. For each of the headings it adds a line item to the list, with the text (`this.text()`) and the page reference of the respective heading.

After the loop, the end tag of the list is added to the variable.

Finally, the script adds the variable - that now contains the table of contents - after the `results`. The `results` object contains the elements that match the selector of the script. So, if the script's selector selects the title of the table of contents, the table of contents will be added after that.

The following style rules, added to the style sheet, will align the chapter titles to the left and the page numbers to the right:

```

#toc li {
text-align:left;
}
#toc span {
float: right;
}

```

Note that these styles use the list's ID, that was defined in the first line of code. For information about style sheets, see "Styling templates with CSS files" on page 553.

parent()

Returns the parents of the elements in a set. (In HTML, a parent is an element that contains another element.)

To get an ancestor that matches a particular selector, use `closest()` (see "Examples" on page 894).

Example

Assume that there are three paragraphs in a Box and that one of those paragraphs matches the selector of this script. The paragraph is stored in the `results` object (see "results" on page 878). The script retrieves the Box (which is the parent of the paragraph) using `results.parent()`, and then changes its background color to red.

```
results.parent().css('background-color', 'red');
```

prepend()

Insert content at the beginning of each element in the set of HTML elements that match the selector of the script or of another query in the template (see "query()" on page 926). See also: "Examples" on page 886.

prepend(content)

Insert content as the first element to each element in the set of HTML elements that match the selector of the script or of another query in the template (see "query()" on page 926). `Append` creates a new result set.

content

HTML string, string or HTML string to insert after the matched elements. In case a plain text string is provided, it is automatically wrapped in a `` element to avoid orphan text nodes to appear in the `<body>` element.

Examples

This script inserts a heading as the first element in an element that has the ID `#box`.

```
results.prepend("<h1>Personal information</h1>");
```

Selector	Matched element	Matched element after script execution
<code>#box</code>	<code><div id="box"> <p>Peter Parker</p> </div></code>	<code><div id="box"> <h1>Personal information</h1> <p>Peter Parker</p> </div></code>

This script inserts a heading as the first element in an element that has the class `name`.

```
results.prepend("<b>Name: </b>");
```

Selector	Matched element	Matched element after script execution
.name	<pre><div> <h1>Personal information</h1> <p class="name">Peter Parker</p> </div></pre>	<pre><div> <h1>Personal information</h1> <p class="name">Name: Peter Parker</p> </div></pre>

This script inserts content in multiple `<div>` elements at the same time.

```
results.prepend("<h1>Personal information</h1>");
```

Selector	Matched element	Matched element after script execution
div	<pre><div id="box"> <p>Peter Parker</p> </div> <div id="box"> <p>Peter Parker</p> </div></pre>	<pre><div id="box"> <h1>Personal information</h1> <p>Peter Parker</p> </div> <div id="box"> <h1>Personal information</h1> <p>Peter Parker</p> </div></pre>

This script prepends a snippet that contains the text "`<h1>Personal information</h1>`".

```
var a = loadhtml('snippets/snippet.html');
results.prepend(a);
```

Selector	Matched element	Matched element after script execution
div	<pre><div id="box"> <p>Peter Parker</p> </div></pre>	<pre><div id="box"> <h1>Personal information</h1> <p>Peter Parker</p> </div></pre>

This script uses the function `query()` to find a box. Then it inserts a heading as the first element in that box.

```
query("#box").prepend("<h1>Personal information</h1>");
```

Matched element	Matched element after script execution
<pre><div id="box"> <p>Peter Parker</p> </div></pre>	<pre><div id="box"> <h1>Personal information</h1> <p>Peter Parker</p> </div></pre>

This script uses the function `query()` to find a box, prepends a heading and sets the text color of the entire box to red.

```
query("#box").prepend("<h1>Personal information</h1>").css(
"color", "red");
```

Matched element	Matched element after script execution
<pre><div id="box"> <p>Peter Parker</p> </div></pre>	<pre><div id="box" style="color: red;"> <h1>Personal information</h1> <p>Peter Parker</p> </div></pre>

Note: the way the functions `prepend()` and `css()` are used in this script is called 'chaining'. Chaining is optional; the same could be achieved by storing the result of the query in a variable:

```
var box = query("#box");
box.prepend("<p>Peter Parker</p>");
box.css("color", "red");
```

remove()

Removes each element in a set from the DOM.

This function returns a new result set containing each removed element. These can be changed and inserted in the document. This could be beneficial in terms of performance, as manipulating elements inside the DOM is relatively time consuming.

Examples

This script removes all Span elements found in the template.

```
results.remove();
```

Selector	Paragraph before script execution	Paragraph after script execution
span	<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit.</p>	<p>Lorem ipsum amet, consectetuer adipiscing elit.</p>

The selector of the following sample script is `tbody`. Before this script runs, the table body consists of a single placeholder row with three cells. After running the script, it contains thirty rows. To improve performance, most of the DOM manipulation takes place on detached elements.

```
// Detach the placeholder row from the DOM
var row = query("tr", results).remove();

// Modify the cells of this row
var cells = row.children();
cells[0].html("some text").css("background-color", "yellow");
cells[1].html("some text").css("font-weight", "bold");
cells[2].html("some text");

// Create a number of copies
var rows = row.clone();
for (var i = 0; i < 30; i++) {
    rows = rows.add(row.clone());
}
```

```
// Attach all copies to the DOM as children of tbody
results.append(rows);
```

removeAttr()

Removes the specified HTML attribute from an element or from each element in a set of elements. To add or change an attribute, use `attr()` (see "Examples" on page 889).

removeAttr(attributeName)

attributeName

String; the name of the attribute.

Examples

This script looks up an email field in a form (which is an `<input>` with the ID `#email1`) and removes its `readonly` attribute.

```
query("#email1").removeAttr('readonly');
```

removeClass()

Removes the specified class from each element in this result set. Has no effect if the class is not present.

removeClass(classname)

classname

String, space separated list of class names.

Examples

This script removes the class name "foo" from all elements in the results that have this class.

```
results.addClass("foo");
```

Selector	Matched element	Matched element after script execution
p	<code><p class="foo">Hello world</p></code>	<code><p>Hello world</p></code>

replaceWith()

Replaces each element in a set of HTML elements.

replaceWith(content)

Replaces each element in a set of HTML elements. Returns the result set.

content

A query result. This can be an HTML string or a result set.

Examples

Replace elements with a snippet

The following script loads a snippet and then replaces the elements matched by the script's selector with the snippet.

```
var snippet = loadhtml('snippets/mysnippet.html');
results.replaceWith(snippet);
```

Replace elements with a set of snippets

The following script loads snippets and adds their elements to a new, empty result set (using `query()`). Then it replaces a placeholder in the template with the set of snippets.

```
var chapters = query();
for ( var i = 1; i <= 4; i++) {
chapters = chapters.add(loadhtml('snippets/Chapter' + i +
'.html'));
}
results.replaceWith(chapters);
```

show()

Shows the elements in a set. To hide elements (again), use the function "hide()" on page 899.

These functions are used by the Conditional Script Wizard, as you can see when you open a Conditional Script and click the **Expand** button; see "Showing content conditionally" on page 613.

Example

This script hides or shows the elements matched by the selector of the script (which are stored in the `results` object), depending on the value of the data field `Country` in the current record.

```
if (record.fields["Country"] == "CANADA") {
  results.show();
} else {
  results.hide();
}
```

text()

text(): String

Returns the text content of the first element in a result set.

Example

This script loads a snippet into a variable and retrieves an element from the snippet using `query()` and `text()`.

```
var mysnippet = loadhtml('snippets/text-root-wrapped.html');
var subject = query("#subject", mysnippet).text();
results.append("<p style='font-weight: bold;'>" + subject +
"</p>");
```

text(value)

Replaces the text content of each element in a result set by the supplied value.

Example

This script loads a snippet, then looks for placeholders using `find()`, and replaces them using `text(value)`.

```
var mysnippet = loadhtml('snippets/snippet.html');
mysnippet.find('@var1@').text('OL Connect 1');
mysnippet.find('@var2@').html('<i>OL Connect 2</i>').css('text-
decoration', 'underline');
results.replaceWith(mysnippet);
```


Each

A generic iterator function, to iterate over the elements in the result set.

each(callback)

Iterates over the elements in a set, such as the enumerable properties of an object, in arbitrary order. For each distinct property, statements can be executed.

callback

A function. The callback function is passed the iteration index and the current element. In the scope of the callback function, `this` refers to the current element.

Examples

The following two scripts demonstrate a simple iteration over the elements in the `results` (the set of HTML elements that match the selector of the script).

This script sets the background color of each of the elements to red. (This is just to demonstrate how this function works. It is easier to change the style of a set of HTML elements using the `css()` function; see "Examples" on page 895.)

```
results.each(function(index) {
    results[index].css('background-color', 'red');
});
```

The following script adds a random integer to each element in the result set.

```
results.each(function(index) {
    var test = Math.floor(Math.random() * 10) + 1);
    this.html(test);
});
```

Selector	Matched element	Matched element after script execution
p	<p></p> <p></p> <p></p>	<p>3</p> <p>1</p> <p>7</p>

This script gets the row index (of the current element in the set) and puts it in a paragraph.

```

results.each(function(index) {
    this.text(index);
})

```

Selector	Matched element	Matched element after script execution
p	<p></p> <p></p> <p></p>	<p>0</p> <p>1</p> <p>2</p>

Using each () in a translation script

The following script first loads a snippet containing translation strings, depending on the value of a field. Then it inserts translations by iterating over elements in the `results` (the set of HTML elements that match the selector of the script) and setting the HTML of each element with a value from the array of translation strings.

```

var strings = loadjson('snippets/' + record.fields.locale +
'.html');
results.each(function(index) {
    if( strings[this.attr('data-translate')] )
        this.html(strings[this.attr('data-translate')]);
});

```

Note: for documentation on the `data-*` attribute, see http://www.w3schools.com/tags/att_global_data.asp.

Selector	Matched element	Matched element after script execution
p	<p data-translate="first"></p> <p data-translate="last"></p> <p data-translate="email"></p>	<p>primero</p> <p>último</p> <p>dirección de correo electrónico</p>

For...in

Can be used to iterate over fields in a data set or rows in detail table. Also see <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/for...in>.

```
for(variable in object) { ... }
```

Iterates over the enumerable properties of an object, in arbitrary order. For each distinct property, statements can be executed.

Examples

This script iterates over field names in the current record and adds them to a paragraph.

```
for(var i in record.fields){  
    results.after("<p>" + i + "</p>");  
}
```

Selector	Matched element	Matched element after script execution
#test	<h1 id="test">Fields</h1>	<h1 id="test">Fields</h1> <p>first</p> <p>last</p> <p>email</p>

This script iterates over fields in the current record, retrieving their values. Then it adds the values to a paragraph.

```
for(var i in record.fields){  
    results.after("<p>" + record.fields[i] + "</p>");  
}
```

Selector	Matched element	Matched element after script execution
#test	<h1 id="test">Fields</h1>	<h1 id="test">Fields</h1> <p>Peter</p> <p>Parker</p> <p>pparker@localhost.com</p>

This script iterates over rows in a detail table and adds the contents of the 'country' field to a paragraph.

```
for(var i in record.tables['countries']) {
    results.after("<p>" + record.tables['countries'][i].fields
['country'] + "</p>");
}
```

Selector	Matched element	Matched element after script execution
#countries	<h1 id="countries">Countries</h1>	<h1 id="countries">Countries</h1> <p>The Netherlands</p> <p>Canada</p> <p>Australia</p>

This script iterates over rows in a detail table and adds the contents of the 'ItemID2' field to an option. The <option> tag defines an option in a select list in an HTML form.

```
for(var i in record.tables['detail']) {
    var str = record.tables['detail'][i].fields["ItemID2"];
    results.append("<option value='" + str + "'" + str +
"</option>");
}
```

formatter

The `formatter` is a global object that allows you to format values in a script.

The Text Script Wizard also allows you to format variable data; see "Using the Text Script Wizard" on page 607 and "Formatting variable data" on page 610.

Note

The `TextFormatter` object is now deprecated and will eventually be removed.

Functions

Function	Description
<ul style="list-style-type: none">• <code>currency()</code>• <code>currencyNoSymbol()</code>• <code>grouped()</code>• <code>integer()</code>• <code>integerUngrouped()</code>• <code>number()</code>• <code>numberUngrouped()</code>	<p>The <code>currency()</code>, <code>grouped()</code>, <code>integer()</code> and <code>number()</code> functions allow you to format a number, possibly with a custom pattern. See "Number functions" on page 923.</p>
<ul style="list-style-type: none">• <code>date()</code>• <code>dateLong()</code>• <code>dateMedium()</code>• <code>dateShort()</code>• <code>dateTime()</code>• <code>dateTimeLong()</code>• <code>dateTimeMedium()</code>• <code>dateTimeShort()</code>• <code>timeLong()</code>• <code>timeMedium()</code>• <code>timeShort()</code>	<p>The <code>date()</code>, <code>dateTime()</code> and <code>time()</code> functions allow you to format a date and/or time in different ways. See "Creating a Date object from a string" on page 917.</p>
<ul style="list-style-type: none">• <code>lowerCase()</code>• <code>upperCase()</code>• <code>properCase()</code>	<p>The text formatting functions are used on Strings. <code>lowerCase()</code> transform all characters to lowercase, <code>upperCase()</code> transforms all characters to uppercase and <code>properCase()</code> transforms the first character of each word to uppercase and all other characters to lowercase.</p>

Date, date/time and time functions

- `date()`
- `dateLong()`
- `dateMedium()`
- `dateShort()`
- `dateTime()`
- `dateTimeLong()`
- `dateTimeMedium()`
- `dateTimeShort()`
- `time()`
- `timeLong()`
- `timeMedium()`
- `timeShort()`

Note

The locale also influences the output of the different Date functions; see "Locale" on page 590.

Tip

To format a date from a `date` field in the record set, you can enter a formatting pattern directly in the Text Script Wizard; see "Using the Text Script Wizard" on page 607, "Formatting variable data" on page 610 and "Date and time patterns" on page 918).

`date(value, pattern)`

Formats a date object using a custom pattern.

value

A Date object. A Date can contain a date and time.

pattern

String. The custom pattern may consist of pattern letters, separating symbols and quoted text, for example: "MMMM dd, yyyy"; see "Date and time patterns" on page 918. Note that the repetition of pattern letters determines the exact presentation.

dateLong(value)

Formats a date as long string representation, for example **April 1, 2016**.

value

A Date object. A Date can contain a date and time.

dateMedium(value)

Formats a date as medium string representation, for example **01/04/16**.

value

A Date object. A Date can contain a date and time.

dateShort(value)

Formats a date as short string representation, for example **1-Apr-2016**.

value

A Date object. A Date can contain a date and time.

dateTime(value, pattern)

Formats a date and time object using a custom pattern.

value

A Date object. A Date can contain a date and time.

pattern

String. The custom pattern may consist of pattern letters, separating symbols and quoted text, for example: "yyyy.MM.dd G 'at' HH:mm:ss z"; see "Date and time patterns" on page 918. Note that the repetition of pattern letters determines the exact presentation.

dateTimeLong(value)

Formats a date and time as long string representation, for example **April 1, 2016 12:00:00 EDT AM**.

value

A Date object. A Date can contain a date and time.

dateTimeMedium(value)

Formats a date and time as medium string representation, for example **1-Apr-2016 12:00:00 AM**.

value

A Date object. A Date can contain a date and time.

dateTimeShort(value)

Formats a date and time as short string representation, for example **01/04/16 12:00 AM**.

value

A Date object. A Date can contain a date and time.

time(value, pattern)

Formats a time using a custom pattern.

value

A Date object. A Date can contain a date and time.

pattern

String. The custom pattern may consist of pattern letters, separating symbols and quoted text, for example: "'at' HH:mm:ss z"; see "Date and time patterns" on page 918. Note that the repetition of pattern letters determines the exact presentation.

timeLong(value)

Formats a time as long string representation, for example **12:00:00 EDT AM**.

value

A Date object. A Date can contain a date and time.

timeMedium(value)

Formats a time as medium string representation, for example **12:00:00 AM**.

value

A Date object. A Date can contain a date and time.

timeShort(value)

Formats a time as short string representation, for example **12:00 AM**.

value

A Date object. A Date can contain a date and time.

Examples

The following script passes the value of a field in the record set to the `date()` function. This will only work if the type of the field has been set to Date in the Data Mapping Configuration and if the field contains a valid date.

```
var myDate = formatter.date(records.fields.DATE, "MM/dd/yyyy");
```

The custom pattern that the script provides, outputs the month and day in two digits each and the year in four digits: 05/21/2016. For more examples of formatting patterns, see "Date and time patterns" on the facing page.

Creating a Date object from a string

In a Data Mapping Configuration you can set the type of a field to Date, but when you open a data file or database in the Designer without a Data Mapping Configuration, all fields are text fields (fields of the type `string`). The `formatter` cannot be used to format a string with a particular date format. The solution is to store the string in a variable as a Date object, and use the `formatter` with that variable.

The following sample script demonstrates this solution. It splits a string into parts and then creates a new Date object with the parts in the correct order. To construct a Date, the parts of the date must be put in the following order: year, month, day, and optionally hours, minutes, seconds, milliseconds (see http://www.w3schools.com/js/js_dates.asp and https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Date.) When the time is omitted, it defaults to 12:00:00 AM.

```
/* Convert the string 21-12-1997 into a valid JavaScript date */
    var strDate = record.fields["date"];
    var dateParts = strDate.split("-");
    var date = new Date(dateParts[2], (dateParts[1] - 1), dateParts
[0]);
```

Note

JavaScript counts months from 0 to 11. January is 0. December is 11.

Another way to put a string in a Date is to use the `Date.parse` function; see https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Date/parse.

The `date` variable can be used as the value in the `date`, `dateTime` or `time` functions of the `formatter`.

```
var myDate = formatter.date(date, "MM/dd/yyyy");
```

Date and time patterns

Dates and times in a template originating from a `date` field in a record set can be displayed using a custom pattern. You can type the pattern directly in the Format field in the Text Script Wizard, lest the field type is set to Date in a Data Mapping Configuration and the field contains a valid date; see "Using the Text Script Wizard" on page 607 and "Formatting variable data" on page 610. In the Script Editor, the pattern can be passed to a `date`, `dateTime` or `Time` function of the `formatter`; see "formatter" on page 912.

The custom pattern may consist of pattern letters (see below), for example: "MM/dd/yyyy". The components can be separated with a space or a symbol, e.g. `.`, `/`, `-`. Text must be put in quotes.

The repetition of pattern letters determines the exact presentation. For example, if the number of pattern letters for a month is less than 3 (M or MM), the month is displayed as a number. If the number of pattern letters is 3 (MMM), it will be displayed as text; if available, a short or abbreviated form of the month's name will be used. If the number of pattern letters is 4 or more (MMMM), the month's full name is displayed.

Note

The pattern letters and patterns on this page are only suitable for **displaying** dates and times in templates, not for extracting dates in the DataMapper module. For pattern letters and patterns available in the DataMapper, see "Date" on page 171.

Pattern letters

Letter	Component	Presentation	Examples
G	Era designator	Text	AD
y	Year	Year	1996; 96
Y	Week year	Year	2009; 09
M	Month in year	Month	July; Jul; 07
w	Week in year	Number	27
W	Week in month	Number	2
D	Day in year	Number	189
d	Day in month	Number	10
F	Day of week in month	Number	2
E	Day name in week	Text	Tuesday; Tue
u	Day number of week (1 = Monday, ..., 7 = Sunday)	Number	1
a	Am/pm marker	Text	PM
H	Hour in day (0-23)	Number	0
k	Hour in day (1-24)	Number	24
K	Hour in am/pm (0-11)	Number	0
h	Hour in am/pm (1-12)	Number	12
m	Minute in hour	Number	30
s	Second in minute	Number	55
S	Millisecond	Number	978
z	Time zone	General time zone	Pacific Standard Time; PST; GMT-08:00
Z	Time zone	RFC 822 time zone	-0800
X	Time zone	ISO 8601 time	-08; -0800; -08:00

zone

Note

These date and time pattern letters and patterns conform to standard Java notation. For more information, see <http://docs.oracle.com/javase/7/docs/api/java/text/SimpleDateFormat.html>.

loadhtml()

Global function that replaces the content (inner html) of each matched element in the result set, alternatively load the data into a variable. The location should be an URL or a relative file path.

Note

Loadhtml() is cached per batch run (based on the URL) in print/email.

loadhtml(location)

Loads all HTML from the specified HTML file.

location

String containing a path that can be absolute or relative to the section/context. Use: snippets/<snippet-name> to retrieve the content from a HTML file residing in the Snippets folder on the Resources panel.

Examples

This script loads a local HTML snippet (from the Resources panel) directly into the matched elements

```
results.loadhtml("snippets/snippet.html");
```

The following script loads a local HTML snippet (Resources panel) into a variable. The `replaceWith()` command is used to replace the element(s) matched by the script's selector with the contents of the snippet.

```
var mysnippet = loadhtml('snippets/snippet.html');  
results.replaceWith(mysnippet);
```

Same result as the previous script, but a different notation:

```
results.replaceWith(loadhtml('snippets/snippet.html'));
```

The following script loads a snippet into a variable and finds/replaces text in the variable before inserting the content into the page. The second find command also adds formatting to the replacing text.

```
var mysnippet = loadhtml('snippets/snippet.html');
mysnippet.find('@var1@').text('OL Connect 1');
mysnippet.find('@var2@').html('<i>OL Connect 2</i>').css('text-decoration', 'underline');
results.replaceWith(mysnippet);
```

This last script loads a snippet into a variable and retrieves an element from the snippet using query().

```
var mysnippet = loadhtml('snippets/text-root-wrapped.html');
var subject = query("#subject", mysnippet).text();
results.append("<p style='font-weight: bold;'>" + subject + "</p>");
```

loadhtml(location, selector)

Retrieves specific content from the specified HTML file.

location

String; the location can be absolute or relative to the section/context. Use: snippets/<snippet-name> to retrieve the content from a HTML file residing in snippets folder of the Resources panel.

selector

String. The supplied selector should conform to CSS selector syntax and allows you to retrieve only the content of matching elements.

Examples

This script loads a **specific element** from a snippet and uses that to replace the `results` (the HTML element or set of HTML elements matched by the selector of the script; see "results" on page 878).

```
var mysnippet = loadhtml('snippets/snippet-  
selectors.html', '#item3');  
results.replaceWith(mysnippet);
```

This script loads the **children** of the selected element.

```
var snippet = loadhtml('snippets/snippet.html', 'foobar').children  
( );  
results.replaceWith(snippet);
```

The next script loads a **remote** snippet, looks for an H1 heading and uses that text.

```
var post = loadhtml('snippets/post.rhtml');  
var h1 = query('h1', post).text();  
results.text(h1);
```

Another example is given in the following how-to: [Using a selector to load part of a snippet](#).

loadjson()

Creates a JSON object based on the text retrieved from the supplied location. The function lets you retrieve content from an JSON enabled server using a standard HTTP request. Popular content management systems, like WordPress (requires JSON API plug-in) and Drupal provide a JSON service/API to retrieve content.

Note

Loadjson() is cached per batch run (based on the URL) in print/email.

This online JSON viewer is handy to debug JSON data: <http://jsonviewer.stack.hu>

loadjson(location)

Loads json data from a remote location.

location

String; the supplied location should be either a URL or a relative file path.

Examples

This sample script retrieves JSON data from a snippet.

```

var localJSON = loadjson('snippets/jsonsnippet.json');
if(localJSON.post){
    results.html("<h3>" + localJSON.post.title + "</h3><p>" +
localJSON.post.modified + "</p>");
}

```

This script retrieves a post from a WordPress site.

```

var wpPost = loadjson('http://192.168.101.58/2013/06/leave-the-
third-dimension-behind-and-focus-on-real-printing-
innovation/?json=1');
if(wpPost.post){
    results.html("<h1>" + wpPost.post.title + "</h1>"
+ wpPost.post.content);
}

```

This script retrieves multiple posts from a WordPress site.

```

var numPosts = 3;
var wpPost = '';
var wpRecentPosts = loadjson('http://192.168.101.58/?json=get_
recent_posts&count=' + numPosts);
if(wpRecentPosts.posts){
    for (var i = 0; i < numPosts ; i++) {
        wpPost += "<p>" + wpRecentPosts.posts[i].title + "</p>";
    }
}
results.after(wpPost)

```

Number functions

Note

The locale also influences the output of some Number functions; see "Locale" on page 590.

Tip

For fields that contain a number, you can also enter a formatting pattern directly in the Text Script

Wizard; see "Using the Text Script Wizard" on page 607, "Formatting variable data" on page 610 and "Number patterns" on the next page).

currency(value)

Formats a number as an amount of money. Which currency symbol and which thousands separator are used depends on the Locale; see "Locale" on page 590.

value

A number. This can be a value from a field that contains a SmallInteger, BigInteger, Float, SmallCurrency or LargeCurrency.

currency(value, pattern)

Formats a number as an amount of money using a custom pattern. Which currency symbol and which thousands separator are used depends on the Locale; see "Locale" on page 590. For available patterns, see "Number patterns" on the next page.

value

A number. This can be a value from a field that contains a SmallInteger, BigInteger, Float, SmallCurrency or LargeCurrency.

pattern

A custom pattern that may consist of symbols; see "Number patterns" on the next page. Note that the repetition of pattern letters plays a part in determining the exact presentation.

currencyNoSymbol(value)

Formats a number as a currency whilst omitting the currency symbol.

value

A number. This can be a value from a field that contains a SmallInteger, BigInteger, Float, SmallCurrency or LargeCurrency.

grouped(value)

Formats a number using a thousands separator. Which separator is used depends on the Locale, see "Locale" on page 590.

value

A number. This can be a value from a field that contains a `SmallInteger`, `BigInteger`, `Float`, `SmallCurrency` or `LargeCurrency`.

Number patterns

Numbers, used in a template and originating from a field in a record set, can be displayed using a custom pattern. You can type the pattern directly in the Format field in the Text Script Wizard; see "Using the Text Script Wizard" on page 607 and "Formatting variable data" on page 610. Note that for this to work, in the DataMapper the field that contains the value must be set to `SmallInteger`, `BigInteger`, `Float`, `SmallCurrency` or `LargeCurrency`.

In the Script Editor, the pattern can be passed to a function of the `formatter`; also see "formatter" on page 912. The custom pattern may consist of pattern characters (see below), a prefix and a suffix.

Note that strings need to be converted to a number before they can be formatted this way.

The repetition of pattern letters determines the exact presentation. For example, the pattern "00000" limits the number to 5 digits and adds leading zeros to any numbers that are not 5 digits long.

Pattern characters

Symbol	Location	Localized?	Meaning
0	Number	Text	Digit
#	Number	Year	Digit, zero shows as absent
.	Number	Year	Decimal separator or monetary decimal separator
-	Number	Month	Minus sign
,	Number	Number	Grouping separator
E	Number	Number	Separates mantissa and exponent in scientific notation. Need not be quoted in prefix or suffix.

;	Subpattern boundary	Number	Separates positive and negative subpatterns
%	Prefix or suffix	Number	Multiply by 100 and show as percentage
\u2030	Prefix or suffix	Number	Multiply by 1000 and show as per mille value
¤ (\u00A4)	Prefix or suffix	Number	Currency sign, replaced by currency symbol. If doubled, replaced by international currency symbol. If present in a pattern, the monetary decimal separator is used instead of the decimal separator.
'	Prefix or suffix	Text	Used to quote special characters in a prefix or suffix, for example, "'###" formats 123 to "#123". To create a single quote itself, use two in a row: "# o'clock".

Source: <http://docs.oracle.com/javase/7/docs/api/java/text/DecimalFormat.html>.

query()

This function creates a new result set, containing the HTML elements in the current section that match the supplied CSS selector. The context (optional) allows you to restrict the search to descendants of one or more context elements.

The new result set is of the type `QueryResults`, just like the `results` object which is also the result of a (hidden) query. All functions that can be used with the `results` object can also be used with this result set; see "results" on page 878.

query(selector)

Creates a new result set containing the HTML elements in the current section that match the supplied CSS selector.

selector

A String containing a CSS selector. See http://www.w3schools.com/cssref/css_selectors.asp for CSS selectors and combinations of CSS selectors.

Examples

Look for an element with a certain ID

This script applies a style rule to the queried elements.

```
query("#test1").css("color", "yellow");
```

Matched element;	Matched element after script execution
<p id="test1">foo</p>	<p id="test1" style="color: yellow;">foo</p>

Look for an element in a snippet

The following script loads a snippet. Then it looks up an element in a snippet and sets its text. Finally, it replaces the elements matched by the script's selector by the snippet.

```
var snippet = loadhtml('snippets/mysnippet.html');  
query("#foo", snippet).text("bar");  
results.replaceWith(snippet);
```

query(selector, context)

Creates a new result set containing the HTML elements that match the supplied CSS selector. The context (optional) allows you to restrict the search to descendants of one or more context elements.

selector

A String containing a CSS selector. See http://www.w3schools.com/cssref/css_selectors.asp for CSS selectors and combinations of CSS selectors.

context

A result set (the result of another query) or an HTML string. If the passed context is not a result set or HTML string it will be coerced to a String and interpreted as HTML.

Examples

This script performs a query in the results of another query.

```
var table = query("table");
var rows = query("tr", table);
var cells = query("td", rows);
```

Since the `results` object also is the result of a query (for elements that match the selector of the script), it can be passed as context. For example in a script with the selector "table".

```
var rows = query("tr", results);
```

query(html)

Creates a new HTML element on the fly from the provided string of raw HTML, and returns the new element.

html

A String containing a HTML element. Tags that can contain other elements should be paired with a closing tag.

Example

The following script adds a paragraph to the `results` (elements that match the selector of the script).

```
results.append("<p>This is a new paragraph.</p>');
```

Tip

The Dynamic Attachment script uses this function to add an attachment to an Email section; see "Email attachments" on page 379.

resource()

The `resource()` function returns information about an image resource. It can also be used to check if a file exists.

This function is useful in a Control Script, for example to check the number of pages or the page height and width before setting it as a background (see "Control Script: Setting a Print section's background" on page 653).

resource(location, pageNumber)

location

The location should be a URL relative to the template root or an absolute file-based URL (without protocol), e.g. "C:/myfile.pdf".

pageNumber (optional)

The desired page. Counting starts at 1. If no page number is given, information about the first page will be retrieved.

The returned object is of the type `ImageInfo`. It has the following fields:

Field	Type	Description
height	float	The height of the current page (in points).
page	Number	Current page number (counting from 1) within the resource.
pages	Number	The total number of pages in the resource.
width	float	The width of the current page (in points)

Examples

This script retrieves the second page of a PDF that is present in the template's resources.

```
var pdf = resource("images/stamp.pdf", 2);
var height = pdf.height;
var width = pdf.width;
var numberOfPages = pdf.pages;
```

In this script, the function is used to check if a file exists.

```
if(resource("C:/paw.pdf")) {
    //exists
} else {
    //oops
}
```

Control Script API

The table below lists the objects that are the most important in Control Scripts. Click through to the object to find a description and sample scripts.

See "Control Scripts" on page 645 for information about this kind of scripts, how to insert them and what you can do with them.

Object	Usage
"section" on page 936	Much of the Control Script magic is performed by setting one of the fields of the <code>section</code> object. Via the <code>section</code> object you can omit, select and clone sections; add a background to a Print section; add a header to an email; etc. A section can be retrieved via the context that it belongs to, using <code>merge.template.contexts.ContextType.sections["section name"]</code> . For example: <code>merge.template.contexts.PRINT.sections["Section EN"]</code> .
"context" on page 932	Object that contains one context and its sections. It is accessed through the <code>template</code> object: <code>merge.template.contexts</code> . To get access to one context, you have to specify the <code>ContextType</code> (see "ContextType" on page 950), for example: <code>var printContext = merge.template.contexts.PRINT;</code> Through the <code>merge</code> object you can find out which context is currently being merged: <code>merge.context</code> .
"template" on page 946	The <code>template</code> object contains all contexts and sections. It is accessed through the <code>merge</code> object: <code>merge.template</code> .
"merge" on page 935	The <code>merge</code> object gives access to the template with all of its contexts and sections .
channel (see "Channel" on page 949)	The channel for which output is generated. This is registered in the <code>merge</code> object: <code>merge.channel</code> . Note that the channel doesn't change when the output consists of different contexts. When generating email, for example, the channel is EMAIL, even when merging the Print context to attach it to the email.
"record" on	The current record in the main data set. To get the value of a field in the

Object	Usage
page 876	record, use <code>record.fields['fieldname']</code> or <code>record.fields.fieldname</code> .

Other objects that are available to Control Scripts

The list above isn't exhaustive: most of the objects listed in the Designer API (see "Designer Script API" on page 874) are also available in Control Scripts. Not all of those objects can be used in Control Scripts, however. This is because Control Scripts differ from template scripts in two ways:

- Control Scripts don't have a selector, like template scripts do. A selector selects parts of the content of a section and stores them in the `results` object, so that they can be modified in the script. As Control Scripts don't have a selector, the `results` object can't be used there. Control Scripts don't touch the content - meaning, the text flow - of the sections.
- Control Scripts run before all other scripts. When a template consists of several contexts, and these contexts are combined in the output - for example, when an Email is generated with the Print context as attachment - all scripts run once for each context, but Control Scripts always go first.

automation

This object (of the type: Automation) encapsulates the properties of the PlanetPress Workflow process that triggered the current operation.

Note

This object is only available in a **Web** context.

Note

The `automation` object available in DataMapper scripts is not of the same type. It has different properties.

Properties

The following table lists the properties of the **Automation** object.

Property	Description
jobInfos	Returns an object containing JobInfo 1 to 9 values from PlanetPress Workflow.
Properties	Returns an object containing additional information (file name, process name and task ID) from PlanetPress Workflow.

Accessing automation properties

To access JobInfo 1 to 9 (defined in Workflow):

```
automation.jobInfos.JobInfo1;
```

To access ProcessName, OriginalFilename or TaskIndex (defined in Workflow):

```
automation.properties.OriginalFilename;
```

Example

Assume that a Workflow process can be triggered when an XML file appears in a certain folder. The XML file contains data that you want to show on a web page.

Add a Set Job Infos and Variables Task to the Workflow process. Define a Job Info (see [Set Job Infos](#)), say, %9, and fill it with data from the XML, for example:

```
xmlget ('/request[1]/values[1]/first[1]', Value, KeepCase, NoTrim)
```

In Connect Designer, you can use the automation object to retrieve the value in a script, like this:

```
var my_var = automation.jobInfos.JobInfo9;
```

context

In a Control Script, the `context` object represents one context in the template.

Which contexts are available in the template can be queried using `merge.template.contexts`.

The context being merged can be queried using `merge.context`.

Field	Type	Description
sections	Array	<p>Array of sections (see "section" on page 936) inside a particular context defined in the template.</p> <p>Note: When using <code>merge.context.sections</code> keep in mind that for example 'Section X' might only exist in your Print context, so using <code>merge.context.sections['Section X']</code> without enclosing it in the <code>if</code> statement <code>if (merge.context.type == ContextType.PRINT) {}</code> will yield an error when the script runs for other contexts.</p> <p>Alternatively, use the <code>template</code> object to access a specific context: <code>merge.template.contexts.PRINT.sections['Section X']</code>.</p>
type	ContextType	The context type: PRINT, EMAIL or WEB (see "ContextType" on page 950).

Example

This script checks if the output channel is EMAIL and if the context to be merged is the Print context (which happens if the Print context is attached to an email). If this is the case, it includes and excludes certain Print sections from the output.

```
if (channel == Channel.EMAIL) {
    if (merge.context.type == ContextType.PRINT) {
        merge.context.sections['Section 1'].enabled = false;
        merge.context.sections['Section 2'].enabled = false;
        merge.context.sections['Section 3'].enabled = true;
    }
}
```

media

The `media` object can be used to specify, enable and position a stationery's front and back in a Control Script (see "Control Scripts" on page 645 and "Control Script API" on page 930).

Note that Media are only used in Print sections (see "Media" on page 353).

The available media are listed in, and retrieved via, the `template` object (see "template" on page 946), for example: `var myMedia = merge.template.media.My_Media.`

Fields

Field	Type	Description
stationery	Stationery	The Stationery's object's <code>front</code> and <code>back</code> fields are used to set the front and the back of a Media; see "front, back" below.

front, back

The `front` and `back` fields of the Stationery object are used to set the front and the back of a Media (see "media" on the previous page).

Both `front` and `back` have the following fields.

Field	Type	Description
enabled	boolean	When enabled, the stationery will be included in the output (Print sections only).
url	String	Specifies the image to use as virtual stationery. For a file named <code>My_Media.pdf</code> , stored inside the template resources, the url would be <code>images/My_Media.pdf</code> . The complete syntax for an external file is: file://<host>/<path> . If the host is "localhost", it can be omitted, resulting in file:///<path> , for example: <code>file:///c:/users/Administrator/Pictures/My_Media.jpg</code> .
page	Number	PDF and TIFF files may count more than one page. Specify the page number to use.
position	"MediaPosition" on page 951	The <code>position</code> is used to place the stationery on the page (absolute, centered, fit to media).
top	Measurement	The vertical offset from the top of the page, used to position

Field	Type	Description
		the stationery (only when absolute positioning is selected). This value can be negative.
left	Measurement	The horizontal offset from the left of the page, used to position the stationery (only when absolute positioning is selected). This value can be negative.

merge

In Control Scripts, the root level instance of the object `merge` is the entry point from where you can query and change the way contexts are merged. It gives access to the template with all its contexts and sections.

For sample scripts, follow the links to the respective objects.

For more information about Control Scripts, see "Control Scripts" on page 645 and "Control Script API" on page 930.

Field	Type	Description
channel	"Channel" on page 949	The final output channel: EMAIL, PRINT or WEB. The channel doesn't change when the output consists of different contexts. When generating an email, for example, the channel is EMAIL, even when merging the Print context to attach it to the email.
"context" on page 932	Context	The context rendered by this merge run. If for one record, different contexts need to be output (for example, when the Print context is attached to an email) a record is merged multiple times: once per context. Per merge run, <code>merge.context</code> shows with which context the record is merged.
"section" on the facing page	Section	In template scripts, this object defines the section that is being merged. Note! In Control Scripts, <code>merge.section</code> is only available when the output channel is WEB. To make sure that it is

Field	Type	Description
		<p>defined, use the following statement: <code>if (merge.channel == Channel.WEB && merge.context.type == ContextType.WEB) { ... }</code>.</p> <p>To retrieve any section in a Control Script, use: <code>merge.template.contexts.ContextType.Section ['Section name'];</code> (for example: <code>merge.template.contexts.PRINT.sections ["Section EN"]</code>).</p>
"template" on page 946	Template	This object contains the template and all of its contexts. It can be used to find out which contexts are available in the template, using <code>merge.template.contexts</code> (see "context" on page 932) and to manipulate the sections in those contexts (see "section" below).

section

The `section` object can be used to query and modify how the section (and the related context) will be outputted. It is one of the most important objects in Control Scripts (see "Control Scripts" on page 645 and "Control Script API" on page 930).

Retrieving a section

A section can be retrieved using `merge.template.contexts.ContextType.sections["section name"]`, for example: `merge.template.contexts.PRINT.sections["Section EN"]`.

A section can also be retrieved via `merge.context.sections['section name']`. Remember, however, that when several contexts need to be merged (for example, when the Print context is attached to an email), the script needs to check if the current context is of the type that contains the desired section (for example: `if (merge.context.type == ContextType.PRINT) {}`). When sections in different contexts have the same name, it is safer to use `merge.template.contexts.ContextType.sections["section name"]`.

Fields

Field	Type	Description
"background" on page 943	Background	Print sections only. Used to set a PDF background on a Print section. See "Control Script: Setting a Print section's background" on page 653 and "BackgroundResource" on page 948.
enabled	boolean	<p>Enables or disables this section for output (see "Examples" on page 939). Note that even if a section is disabled, the <code>part</code> and <code>restartPageNumber</code> fields are still effective to define the parts division and page numbering over multiple sections when applicable.</p> <p>The default enabled state for sections (before any control script runs) is as follows: For Web channel requests, the requested web section is enabled by default. It is possible to redirect to another section by disabling the requested section and enabling another section. For Email channel requests on the Web context, only the default section is enabled by default. It is possible to enable different or multiple sections, to control which sections will be attached to the email. For Email channel requests on the Print context all Print sections are enabled by default. It is possible to enable different or multiple sections to control which sections will be attached to the email. For Print channel requests on the Print context all sections are enabled by default.</p>
headers	String	Email sections only. Used to set custom email headers. For examples, see "Adding custom ESP handling instructions" on page 978.

Field	Type	Description
name	String	Used to get or set the name of the section. Note that section names must be unique and that sections cannot have an integer as its name. The name should always include alphanumeric characters. To rename email attachments, use the field <code>part</code> .
ownerPassword	String	Print sections only. Used to set the owner password for a PDF attachment.* Setting only the owner password creates a secured PDF that can be freely viewed, but cannot be manipulated unless the owner password is provided. (Note that the recipient needs Adobe Acrobat to do this, because the Acrobat Reader does not allow users to enter the owner password.) See "Control Script: Securing PDF attachments" on page 658.
part	String	Name for the part. <code>part</code> is used to specify where a new part starts and the title for the part. This is used to split Email attachments. The Email output can, for example, attach 3 PDFs generated from the Print context. The part name will be used as the file name for the attachment. See "Parts: splitting and renaming email attachments" on page 651.
password	String	Print sections only. Used to set the user password and owner password for a PDF attachment to the same value. See "Control Script: Securing PDF attachments" on page 658.*
restartPageNumber	boolean	Print sections only. Enables or disables a restart of the page numbering. When generating Print output this can be used to let page numbering continue over multiple sections. The default value is <code>false</code> , meaning that each section will start with page 1 (to emulate behavior of

Field	Type	Description
		previous versions).

*The password(s) should be set on the first Print section when producing a single attachment, or on the first section of each part when producing multiple attachments. Each of the parts (attachments) may have a different (or no) set of passwords.

Passwords set in the Control Script override the password set through the Email PDF password script (see "Email PDF password" on page 378). This allows you to change or remove the password from a specific part. Removal is done by setting the `password` field to `null` or "" (empty string).

Functions

Note

For cloned sections, functions are not available.

Function	Description
"Examples" on page 944	Clone this section. See "Dynamically adding sections (cloning)" on page 655.
addAfter()	Add a cloned section after this section.
addBefore()	Add a cloned section before this section.

Examples

Conditionally skipping or printing Print sections

This script disables all Print sections and then re-enables one of them, depending on a value in the current record.

```
var printSections = merge.template.contexts.PRINT.sections;
printSections['Section EN'].enabled = false;
printSections['Section FR'].enabled = false;
```

```

if(record.fields.Language === 'FR'){
    printSections['Section FR'].enabled = true;
} else {
    printSections['Section EN'].enabled = true;
}

```

Selecting different sections for Print output and Email PDF attachment

This script selects a different Print section for output, depending on the output channel (Email or Print).

```

var printSections = merge.template.contexts.PRINT.sections;

if(merge.channel === Channel.EMAIL) {
    printSections['Section 1'].enabled = false;
    printSections['Section 2'].enabled = true;
}

if(merge.channel === Channel.PRINT) {
    printSections['Section 1'].enabled = true;
    printSections['Section 2'].enabled = false;
}

```

Setting the name of Email PDF attachments

This script renames the file name of an attachment by setting the part name of a section (see "Parts: splitting and renaming email attachments" on page 651).

```

var section = merge.template.contexts.PRINT.sections['Section 1'];
section.part = 'Invoice ' + record.fields['InvoiceNo'];

```

Controlling multiple Email attachments

The following script attaches the following sections to an email:

- Print section 3 + 4 as attachment with continued page numbers
- Print section 6 as separate attachment (also see "Parts: splitting and renaming email attachments" on page 651)
- Web sections A and B as separate attachment

```

if (channel == Channel.EMAIL) { // only when generating Email
output
if (merge.context.type == ContextType.PRINT) {

```



```

merge.context.sections['Section 1'].enabled = false;
merge.context.sections['Section 2'].enabled = false;
merge.context.sections['Section 3'].enabled = true;
merge.context.sections['Section 3'].part = "PDFAttach1";
merge.context.sections['Section 4'].enabled = true;
merge.context.sections['Section 4'].restartPageNumber = false;
merge.context.sections['Section 5'].enabled = false;
merge.context.sections['Section 6'].enabled = true;
merge.context.sections['Section 6'].part = "PDFAttach2";
} else if (merge.context.type == ContextType.WEB) {
merge.context.sections['default Section'].enabled = false; //
disable whatever is the default section
merge.context.sections['Section A'].enabled = true;
merge.context.sections['Section A'].part = "WebPartA";
merge.context.sections['Section B'].enabled = true;
merge.context.sections['Section B'].part = "WebPartB";
}
}

```

Note

For another example, see this how-to: [Output sections conditionally](#).

Note

If the Email PDF Password Script Wizard defines a password, and a template has a Control Script that creates multiple PDF attachments, all the attachments are secured by the same password by default. Using a Control Script, you can set set different passwords for attachments; see "Control Script: Securing PDF attachments" on page 658.

Positioning the background of a Print section

These scripts both set the background of a Print section to the same PDF, but they position it differently.

Using absolute positioning

```

var activeSection = merge.template.contexts.PRINT.sections['Section
1'];

```

```
activeSection.background.source = BackgroundResource.RESOURCE_PDF;
activeSection.background.position = MediaPosition.ABSOLUTE;
activeSection.background.left = "10mm";
activeSection.background.top = "10mm";
activeSection.background.url = "images/somepage.pdf";
```

Scaling to Media size

```
var activeSection = merge.template.contexts.PRINT.sections['Section
1'];
activeSection.background.source = BackgroundResource.RESOURCE_PDF;
activeSection.background.position = MediaPosition.FIT_TO_MEDIA;
activeSection.background.url = "images/somepage.pdf";
```

See also: "BackgroundResource" on page 948, "MediaPosition" on page 951 and "Control Script: Setting a Print section's background" on page 653.

Cloning Print sections

For background information on cloning Print sections, see: "Dynamically adding sections (cloning)" on page 655.

Cloning a section based on the number of records in a detail table

This script creates as many clones of a section as there are records in a detail table. It assigns the new sections a unique name.

```
var printSections = merge.template.contexts.PRINT.sections;
var numClones = record.tables['detail'].length;
for( var i = 0; i < numClones; i++){
    var clone = printSections["Section 1"].clone();
    clone.name = "my_section_clone_" + i;
    printSections["Section 1"].addAfter(clone);
}
```

Cloning a section based on data and assign a background PDF

This script clones a section based on data fields. It disables the source section first and then calls the `addPolicy` function. `addPolicy` clones the section, renames it and sets a PDF from the resources as its background. It explicitly enables the clone and then adds it to the Print context.

```
var printSections = merge.template.contexts.PRINT.sections;
merge.template.contexts.PRINT.sections["Policy"].enabled = false;
if(record.fields.policy_a == 1) {
```

```

        addPolicy('a');
    }
    if(record.fields.policy_b == 1) {
        addPolicy('b');
    }
    function addPolicy(policy){
        var resourceUrl = 'images/policy-' + policy + '.pdf';
        var clone = printSections["Policy"].clone();
        clone.name = "policy_" + policy;
        clone.background.url = resourceUrl;
        clone.enabled = true;
        printSections["Policy"].addAfter(clone);
    }
}

```

background

The `background` object holds the PDF background of a Print section (see "section" on page 936). For examples, see "Control Script: Setting a Print section's background" on page 653.

Fields

Field	Type	Description
allPages	Boolean	Show all pages from the PDF.
end	Number	The end page of the PDF to use as a background for the section.
left	Measurement	The left offset of the PDF background (only when absolute positioning is selected).
position	"MediaPosition" on page 951	Set the position of the PDF background (Absolute, centered, fit to media).
source	"BackgroundResource" on page 948	Set the source of the PDF background (None, Datamapper, PDF Resource).
start	Number	The start page of the PDF to use as a background for the section.

Field	Type	Description
top	Measurement	The top offset of the PDF background (only when absolute positioning is selected).
url	String	The location of the PDF to use as a background for the section. For a file named background.pdf, stored inside the template resources, the url would be <code>images/background.pdf</code> . The complete syntax for an external file is: <code>file://<host>/<path></code> . If the host is "localhost", it can be omitted, resulting in <code>file:///<path></code> , for example: <code>file:///c:/resources/images/image.jpg</code>

clone()

This function returns a new set containing a copy of each element in a set; see "Dynamically adding sections (cloning)" on page 655.

Note

Due to resource constraints, the number of unique clones that can be created throughout a job is limited to around 20. A clone is considered unique if it has a different name. This is a rough estimate; if the template is simple, up to 60 clones may be created.

The limit only applies to the amount of unique clones. There is no limit to the amount of `clone()` function calls.

To duplicate an existing template element, clone it before calling `append()`; see "Examples" on page 886.

Examples

This script performs an iteration over the elements in the `results` (the elements that match the selector of the script).

```
var row = query("tbody tr", results).clone();
query("tbody", results).append(row);
```

The following script clones an existing table row to match the number of rows in a detail table. Afterwards it iterates over the rows to populate the fields.

```
// Create the number of rows based on the records in the detail
table
// We start at 1 so the boilerplate row is used too and there is no
need to delete that row
for(var r = 1; r < record.tables['detail'].length; r++) {
results.parent().append(results.clone());
}

// Iterate over the rows and populate them with the data from the
accompanying data row
query("#table_2 > tbody > tr").each(function(i) {
this.find('@ItemNumber@').text( record.tables['detail'][i].fields
["ItemNumber"]);
this.find('@ItemOrdered@').text( record.tables['detail'][i].fields
["ItemOrdered"]);
this.find('@ItemTotal@').text( record.tables['detail'][i].fields
["ItemTotal"]);
this.find('@ItemDesc@').text( record.tables['detail'][i].fields
["ItemDesc"]);
this.find('@nr@').text(i);
});
```

The following script clones and populates a boilerplate row. Once completed you will need to hide the boilerplate row.

```
for(var i = 0; i < record.tables['detail'].length; i++) {

var row = results.clone(); //Clone our boilerplate row

row.find('@ItemNumber@').text( record.tables['detail'][i].fields
["ItemNumber"]);
row.find('@ItemOrdered@').text( record.tables['detail'][i].fields
["ItemOrdered"]);
row.find('@ItemTotal@').text( record.tables['detail'][i].fields
["ItemTotal"]);
row.find('@ItemDesc@').text( record.tables['detail'][i].fields
["ItemDesc"]);
row.find('@nr@').text( i );

results.parent().append(row);
```

```

}

// Hide our boilerplate row (note that this doesn't really delete
the row).
results.hide();

```

template

The `template` object represents the template with all its contexts and sections. It is used frequently in Control Scripts (see "Control Scripts" on page 645 and "Control Script API" on page 930).

It is retrieved via the merge object: `merge.template` (see "merge" on page 935).

Which contexts are available in the template can be queried using `merge.template.contexts`. To get access to a specific context, you have to specify the `ContextType` (see "ContextType" on page 950).

Field	Type	Description
contexts	Array	Array of contexts (see "context" on page 932) available in the template. The contexts contain the sections (see "section" on page 936).
"media" on page 933	Array	Media available to this template (see "Media" on page 353). For each of them you can specify, enable and position the stationery's front and back.
"properties" on the next page	Properties	This object contains all default properties of the template as well as any custom properties. (On the menu, select File > Properties to view and complement the file properties. See File Properties).

Example

The following Control Script retrieves two Print sections. Then, depending on a value in the current record, it enables one section or the other, so that only one of the two sections appears

in the output.

```
var printSections = merge.template.contexts.PRINT.sections;
printSections['Section EN'].enabled = false;
printSections['Section FR'].enabled = false;

if(record.fields.Language === 'FR'){
    printSections['Section FR'].enabled = true;
} else {
    printSections['Section EN'].enabled = true;
}
```

properties

The `properties` object inside the `template` object (see "template" on the previous page) contains all default properties of the template file as well as any custom properties.

To view and complement the file properties, select **File > Properties** on the menu. See File Properties.

Following are the default properties.

Field	Type	Description
application	String	Application version when the document was last saved
author	String	Original author of the document
company	String	company name
created	Date	Date and time on which the document was created
customProperties	List	A list of defined custom properties
description	String	Description of the document

Field	Type	Description
file	String	File name of the document
keywords	String	Semicolon-separated list of keywords
modified	Date	Date and time on which the document was last saved

Example

This script stores a default property (author) and a custom property in variables.

```
var author = merge.template.properties.author;
var myProperty = merge.template.customProperties.myPropertyName;
```

BackgroundResource

BackgroundResource is an enumeration for the types of background resources for a Print section (see "background" on page 943 and "section" on page 936).

A Print section can be retrieved in script using

```
merge.template.contexts.ContextType.sections["section name"], for example
merge.template.contexts.PRINT.sections["Section EN"].
```

Field	Description
DATAMAPPER_PDF	A PDF file retrieved via the active Data Mapping Configuration. This can be the PDF file that was used as input file, or another type of input file, converted to PDF.
NONE	No PDF background.
RESOURCE_PDF	A PDF file stored in the template or on the network. Note that it isn't possible to use a remotely stored PDF file as a section's background.

Example

The following script sets the background for a section called 'Policy' to `RESOURCE_PDF` and specifies a path for it, using a data value:

```
// Enable the section background and specify that the PDF should be read
// from a resource file rather than using a PDF DataMapper
background
merge.template.contexts.PRINT.sections['Policy'].background.source
= BackgroundResource.RESOURCE_PDF;

// Specify the path
var resourceUrl = 'images/policy-' + record.fields.policy + '.pdf';
merge.template.contexts.PRINT.sections['Policy'].background.url =
resourceUrl;
```

Note

To learn how to set a PDF file as a background image on a Print section without a Control Script, see "Using a PDF file as background image" on page 339.

Channel

`Channel` is an enumeration for the output channels. The active output channel is registered in `merge.channel`.

The channel doesn't change when the output consists of different contexts. When generating email, for example, the channel is `EMAIL`, even when merging the Print context to attach it to the email.

Value	Description
EMAIL	The merge request is for output to Email.
PRINT	The merge request is for output to Print.
WEB	The merge request is for output to Web.

Value	Description
THUMBNAIL	The merge request is for generating a template preview.

Example

The following Control Script selects different sections for Print output and for Email with the Print context attached to it.

```
var printSections = merge.template.contexts.PRINT.sections;

if(merge.channel === Channel.EMAIL) {
    printSections['Section 1'].enabled = false;
    printSections['Section 2'].enabled = true;
}

if(merge.channel === Channel.PRINT) {
    printSections['Section 1'].enabled = true;
    printSections['Section 2'].enabled = false;
}
```

ContextType

ContextType is an enumeration for the context types.

The type of the context that is going to be merged next can be retrieved via `merge.context.type`.

The context type needs to be specified when retrieving a section with `merge.template.contexts.ContextType.sections["section name"]`, for example `merge.template.contexts.PRINT.sections["Section EN"]`.

Value	Description
HTML_EMAIL	The context is the Email context.
PRINT	The context is the Print context.
WEB	The context is the Web context.

Example

This script retrieves two Print sections. Then, depending on a value in the current record, it enables one section or the other, so that only one of the two sections appears in the output.

```
var printSections = merge.template.contexts.PRINT.sections;
printSections['Section EN'].enabled = false;
printSections['Section FR'].enabled = false;

if(record.fields.Language === 'FR'){
    printSections['Section FR'].enabled = true;
} else {
    printSections['Section EN'].enabled = true;
}
```

MediaPosition

In a Control Script, the `position` is an enumeration for the position of background resources for a Print section. It is retrieved and set via `background.position`.

Field	Description
ABSOLUTE	Places the PDF at a specific location on the page. Set the background's top (<code>background.top</code>) and left (<code>background.left</code>) measured from the top and left side of the section.
CENTERED	Centers the PDF on the page, vertically and horizontally.
FIT_TO_MEDIA	Stretches the PDF to fit the page size.

Examples

This script applies **absolute positioning** to the background of a Print section.

```
var activeSection = merge.template.contexts.PRINT.sections['Section 1'];
activeSection.background.source = BackgroundResource.RESOURCE_PDF;
activeSection.background.position = MediaPosition.ABSOLUTE;
```

```
activeSection.background.left = "10mm";  
activeSection.background.top = "10mm";  
activeSection.background.url = "images/somepage.pdf";
```

The next script scales the background of a Print section to the size of the **Media**.

```
var activeSection = merge.template.contexts.PRINT.sections['Section  
1'];  
activeSection.background.source = BackgroundResource.RESOURCE_PDF;  
activeSection.background.position = MediaPosition.FIT_TO_MEDIA;  
activeSection.background.url = "images/somepage.pdf";
```

Generating output

When merged with a record set, the templates made in the Designer can generate three types of output: Print, Email and Web.

Print output

Print templates, also called Print *sections*, are part of the Print context. They are meant to be printed to a printer or printer stream, or to a PDF file (see "Generating Print output" on page 956).

The Print context can also be added to Email output as a PDF attachment; see "Generating Email output" on page 973. When generating output from the Print context, each of the Print sections is added to the output document, one after the other in sequence, for each record.

To dynamically select a section for output, use a Control Script; see "Control Scripts" on page 645.

There is a number of settings in the Print context and Print sections that have an impact on how the Print context is printed; see "Print settings in the Print context and sections" on page 333.

To split the Print output into several files, see "Splitting printing into more than one file" on page 961.

Email output

The Email context outputs HTML email with embedded formatting to an email client through the use of an email server. The HTML generated by this context is meant to be compatible with as many clients and as many devices as possible.

Although the Email context can contain multiple Email templates, only one of them can be merged with each record. Which one is used, depends on a setting; see "Email output settings in the Email context and sections" on page 974.

Email Output can be generated in two different ways: from the Designer or via Workflow. In both cases, email is sent in a single batch for the whole record set.

To test a template, you can test the scripts (see "Testing scripts" on page 632) and send a test email first (see Send Test Email), before actually sending the email (see "Generating Email output" on page 973).

Attachments

Output, generated from an Email template, can have the following attachments:

- The contents of the Print context, in the form of a single PDF attachment.
- The output of the Web context, as an integral HTML file.
- Other files, an image or a PDF leaflet for example.

Attaching the Print context and/or the Web context is one of the options in the Send (Test) Email dialog.

To learn how to attach other files, see "Email attachments" on page 379.

Web output

The Web context outputs an HTML web page that contains the HTML text and all the resources necessary to display it.

Web output can be generated in two different ways: it can be attached to an Email template when generating Email output (see above), or it can be generated using Workflow; see "Generating Web output" on page 981.

Although the Web context can contain multiple Web pages, only one of them can be merged with each record. Which one is used, depends on a setting; see "Web output settings in the Web context and sections" on page 983.

Optimizing a template

Scripts

In the process of output generation, the execution of scripts may take up more time than necessary. To optimize a template, it helps to disable scripts that don't have an effect on the context that you're generating output from; see "Managing scripts" on page 629.

Other ways to speed up script execution are described in another topic: "Optimizing scripts" on page 636.

Images

When a template that contains lots of images is merged with a large record set, the many file requests may slow down the process of output generation. The solution is simple: combine the images into a single image file and display the part that holds the image. This reduces the number of file requests and can improve the output speed significantly.

Step 1. Create a file that contains a collection of images.

Static images may go in any type of image file. Store images that need be added dynamically to the template, in one PDF file, one image per page.

There are several tools to combine image files into a single PDF. **ImageMagick** is one of them. You could use the `convert` command of the ImageMagick library:

```
convert C:/myimages/*.jpg C:/myimages/image-collection.pdf
```

You could also use **Connect Designer** itself: create a print template with the size of your images and set the page margins to 0. Create a script that loops over your images and adds them to the text flow of the template. Subsequently generate PDF output and use the resulting file as your collection file.

Step 2. Add the file that contains the collection of images to the template's Resources (see "Adding images" on page 539).

Step 3. Display part of the collection file as an image in the template.

- **Static images** that are part of an image file can be displayed via Cascading Style Sheets (CSS). This technique is much used in web design. In this technique, the file that contains a collection of images is called an **image sprite**. The trick is to create a Box (or Div) for each image and give that box an ID (see "Boxes" on page 513). Then use the ID in a style sheet to select the Box and write a style rule (see "Styling templates with CSS files" on page 553) that sets its background image to the image sprite and positions the image. For an explanation and examples of this style rule, see http://www.w3schools.com/css/css_image_sprites.asp.
- **Dynamically added images** are loaded in a script. To retrieve one page from a PDF file

in a script, add the page parameter to the file path and set that as the source of the image. Here is an example (assuming that the page number is stored in a variable `pageNumber`):

```
var imageStr = "";
var imagePath = "file:///C:/image-collection.pdf?page=" +
pageNumber;
imageStr += ' Print...** allows the following printing options:
  - Using the **Default** output settings.  
For more details, see "Print Using Standard Print Output Settings" on page 958
  - Using the same settings that were **last used** to produce printed output.  
For more details, see "Print Using Standard Print Output Settings" on page 958
  - Using **entirely new output settings** set via the Advanced option, which allows selection from a myriad of print output options.

### Note

These settings cannot be saved for later re-use. To do that, one should instead create printing Presets, which are designed to allow just this behaviour.

For a detailed description see "Print Using Advanced Printer Wizard " on page 959.



- Using previously saved **Printing Preset** options.  
See "[Job Creation Presets](#)" on page 840 and [Output Creation Presets](#) for more details.
- **File > Proof Print...** allows either the default output settings; the last used output settings or previously saved output Presets.  
For more information on this option see "Print Using Standard Print Output Settings" on the facing page. for more details.

## Saving Printing options in Print Presets

Selecting **File > Print Presets** allows you to create or modify printing Presets (which contain all the printing options), which can be saved for re-use in later print runs. This can be particularly handy when creating special print runs, that need to be run periodically.

These presets make it possible to do such things as filtering and sorting records, grouping documents and splitting the print jobs into smaller print jobs, as well as the more standard selection of printing options, such as binding, OMR markings and the like.

See "[Job Creation Presets](#)" on page 840 and "Output Creation Settings" on page 850 for more details.

## Connect Printing options that cannot be changed from within the Printer Wizard

There are a number of settings for the Print context and Print sections that have an impact on how Print sections are printed, which **cannot** be influenced through either a Job Creation Preset or an Output Creation Preset.

These settings are:

- **Duplex printing.** Duplex printing has to be enabled for a Print section, in order to print that section on both sides of the paper. The same applies to Mixplex printing.  
See "Enabling double-sided printing (Duplex, Mixplex)" on page 342.
- **Finishing.** The Print context , as well as each of the Print sections, can have its own **Finishing** settings. In printing, Finishing is the way pages are bound together after they

are printed.

See "Setting the binding style for the Print context" on page 334 and "Setting the binding style for a Print section" on page 341.



Also see "Finishing Options" on page 841 for an explanation of the Finishing options.

- **Bleed.** The margins *around* a page are called the Bleed. It can be used on some printers to ensure that no unprinted edges occur in the final trimmed document. See "Page settings: size, margins and bleed" on page 344.

## Print Using Standard Print Output Settings

When using the **File > Print...** option, the Print Configuration dialog appears. This dialog allows you to print the template using **Default** printer settings, or the **Last Used** printer settings or by using previously created Printing Presets.

To learn how to create Printing Presets please see [Job Creation Presets](#) and [Output Creation Presets](#).

- **Configuration Selection Group:**
  - **Output Creation:** Use the drop-down to select existing Output Creation Presets. Use the Gear button to edit the currently selected Preset or to reload the list of Presets from the system. 
  - **Job Creation:** Use the drop-down to select existing Job Creation Presets. Use the Gear button to edit the currently selected Preset or to reload the list of Presets from the system. 
  - **Preset Summary:** Displays a summary of the settings for the currently selected Presets.

### Note

The Default output type of PDF Output is actually a built in system Preset, whilst the Last Used settings can likewise be considered an un-named and un-saved Preset.

- **Records Group:**

- **All:** Outputs all records in the active dataset.
- **Selection:** Allows selection of a range of records or a custom selection. You can specify individual records separated by semi-colons (;) or ranges using dashes. For example: 2;4;6-10 would print pages 2, 4, 6, 7, 8, 9 and 10.
- **Apply filtering and sorting to record selection** checkbox: Check to filter and/or sort records. Selecting this will open both the "Data Filtering Options" on page 843 and "Sorting Options" on page 844 pages.
- **Copies Group :**
  - **Copies:** Enter the number of output copies you want.
  - **Collate:** When printing multiple copies you can check this checkbox to have the record copies printed together. For example in a three record job the records would print out as 1-1-2-2-3-3, rather than 1-2-3-1-2-3.

## Wizard navigation buttons

- **Advanced** button: Click to open the "Print Using Advanced Printer Wizard " below where you can manually change the printing options.

### Note

Any settings made within the **Advanced Print Wizard** do not permanently update any Preset(s) being used.

- **Print** button: Click to produce print output according to the current settings.
- **Cancel** button: Cancels the Print Wizard, without creating any printout.

## Print Using Advanced Printer Wizard

The **Advanced Printer Wizard** allows you to select from any and all output settings.

The Wizard can be used to generate once-off print runs (either entirely from scratch, or based upon selected pre-existing Presets).

**Note:** These print runs cannot be saved as presets and can only be replicated in the following print run, using the **Last Used** option.

The output settings are determined by selections made throughout the Wizard. For example, if you want to add Inserter Marks to the output, you select the Add Inserter Marks option on the first page of the Wizard, and the Inserter Options page will then appear later in the Wizard.

The first page of the Advanced Printer Wizard is the "Print Options" on page 851 page.

## Adding print output models to the Print Wizard


Connect comes with several pre-prepared print output models. These include Printer Control Language (PCL), Portable Document Format (PDF) and PostScript (including the PostScript variants of PPML, VIPP and VPS).

To keep the Print Wizard interface manageable only a limited range of print output models are available by default. Additional print output models can be added to the list at any time, though. They can be selected from the range of pre-prepared models that come with Connect. The following topic describes how to do this.

After they have been added, the newly selected print output models will be available in the Print Wizard thereafter.

### How to add print output models from within the Print Wizard

Here is how to add print output options from within the Print Wizard dialog itself.

1. Select **File > Print...** from the menu. The Print dialog will be launched.
2. Click on the **Advanced** button. The Print Wizard will be launched.
3. Click the settings button  at the end of the Model selection.
4. Select **Edit available printers** from options.
5. In the **Preferences** dialog, select the print output models to be added to the Print Wizard, then click OK.

### How to add print output models from within the Designer

Here is how to add print output models from within the main Designer interface itself.

1. Select **Window > Preferences...** from the menu. Preference dialog is launched.
2. Select **Print > Available Printers** from the options.
3. In the **Available Printers** area, select the print output options to be added to the Print Wizard, then click OK.

## Splitting printing into more than one file

By default, when Connect saves the print output spool file to a directory, it creates one spool file that contains all the generated documents. It is, however, possible to output one spool file per document, or to create groups of documents and store those in separate spool files.

Where the output should go, and how documents should be grouped, is set in a *Job Creation Preset*.

To make one document or a group of documents go into a separate file, the print job needs to be 'separated'. Separation is one of the options to set in an *Output Creation Preset*.

See "[Generating Print output](#)" on page 956 for a further explanation about Job Creation Presets and Output Creation Presets.

## Print output variables

In Print output, File name variables can be used to create dynamic output file names, while Content variables can be used in additional content and in the Conditional field when selectively adding inserts in the "Inserter Options" on page 865.

### Note

When using a variable in a condition, do not wrap it in  $\$ \{ \}$ . This notation is only necessary in the context of a text (like file names or additional page content).

## File name variables

File name variables are available in a few places in the "Print Options" on page 851:

- In the **Job Output Mask** field when using the Directory option
- In the **Job Name** field when using the Windows Printer option.

If the output is to be separated into multiple files, some Output content variables can also be used as Output file variables, depending on the level of separation (see "Content variables" on page 965).

## Template

### Note

The variables `${template}`, `${template.base}` and `${template.name}` are only available when printing directly from the Designer.

|                                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><code>\${template}</code></b>      | <p><code>\${template}</code> is a shorthand for <code>\${template.base}_</code><br/><code>\${template.nr,0000}.\${template.ext}</code>.</p> <p>It expands to a name based on the template name. A four digit sequence number is added at the end of the base name. The file extension is determined by the selected output format.</p> <p>The <code>0000</code> in <code>\${template.nr,0000}</code> is a format pattern that takes care of formatting the number with at least four digits and leading zero's. See "Formatting date and number values " on page 969, below.</p> <p><b>Example</b><br/>If the template file is <code>C:\Data\My-Invoices-EN.OL-template</code> which gets printed to PDF, then <code>\${template}</code> expands to <code>My-Invoices-EN_0001.pdf</code>.</p> |
| <b><code>\${template.base}</code></b> | <p>Returns the base name of the template, which is the name of the template file without its path and without the trailing file extension.</p> <p><b>Example</b><br/>If the template file is <code>C:\Data\My-Invoices-EN.OL-template</code>, then</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

|                                              |                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                              | <p><code>\${template.base}</code> expands to <code>My-Invoices-EN</code> .</p>                                                                                                                                                                                                                                                                                                                                 |
| <p><b><code>\${template.name}</code></b></p> | <p>Returns the name of the template file without the path.</p> <p><b>Example</b><br/> If the template file is <code>C:\Data\My-Invoices-EN.OL-template</code>, then <code>\${template.name}</code> expands to <code>My-Invoices-EN.OL-template</code>.</p> <p>Note that <code>\${template.name}</code> still includes the extension of the template file (<code>.OL-template</code> in the example above).</p> |
| <p><b><code>\${template.nr}</code></b></p>   | <p>An automatic sequence number belonging to the current output file. It is automatically incremented for each new output file that gets created when Separation has been selected in the Output Creation preset.</p> <p>It is possible to format the number using a pattern and locale. See "Formatting date and number values " on page 969, below.</p>                                                      |
| <p><b><code>\${template.ext}</code></b></p>  | <p>The extension that corresponds to the chosen output format (in lower case).</p> <p>For example, for PDF output, <code>\${template.ext}</code> would be <b>pdf</b>, for PostScript output, <code>\${template.ext}</code> would return <b>ps</b>.</p> <p>Note that <code>\${template.ext}</code> does not include a leading dot.</p>                                                                          |

## File

|                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b><code>\${file}</code></b></p> | <p><code>\${file}</code> is a shorthand for <code>\${file.base}_</code><br/> <code>\${file.nr,0000}.\${file.ext}</code>. It expands to an internally generated file name with a four digit sequence number at the end. The file extension is determined by the selected output format.</p> <p>The <code>0000</code> in <code>\${file.nr,0000}</code> is a format pattern that takes care of formatting the number with at least four digits including leading zero's. See "Formatting date and number values " on page 969, below.</p> |
|-------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                                        |                                                                                                                                                                                                                                                                                                                            |
|----------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                        | <p><b>Example</b></p> <p>When printing to PDF, <code>\${file}</code> could expand to <b>merged.5852941188491153960_0001.pdf</b>.</p>                                                                                                                                                                                       |
| <b><code>\${file.base}</code></b>      | Expands to a unique, internally generated file name, without a trailing dot or extension.                                                                                                                                                                                                                                  |
| <b><code>\${file.ext}</code></b>       | <p>The extension that corresponds to the chosen output format, for example <b>pdf</b> or <b>ps</b>.</p> <p>Note that <code>\${file.ext}</code> does not include a leading dot.</p>                                                                                                                                         |
| <b><code>\${file.nr}</code></b>        | <p>An automatic sequence number belonging to the current output file. It is automatically incremented for each output file that gets created within the same job.</p> <p>It is possible to format the number using a pattern and locale. See "Formatting date and number values " on page 969, below.</p>                  |
| <b><code>\${file.pageCount}</code></b> | <p>This variable was introduced for use in Printer Definitions for PostScript printers.</p> <p>As it entails page buffering and could easily lead to Out of Memory errors with big jobs, <b>usage of this variable in an Output Preset or in the Print Wizard is discouraged; it should be regarded as deprecated.</b></p> |

## Job

|                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><code>\${job}</code></b> | <p><code>\${job}</code> is a shorthand for <code>\${job.base}_\${job.nr,0000}.\${job.ext}</code>. It expands to a name based on the name of the applied Job Creation preset (or 'Untitled' if no Job Creation preset was used). A four digit sequence number is added at the end of the base name. The file extension is determined by the selected output format.</p> <p>The <code>0000</code> in <code>\${job.nr,0000}</code> is a format pattern that takes care of formatting the number with at least four digits including leading zero's. See "Formatting date and number values " on page 969, below.</p> |
|-----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|



|                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-----------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b><code>#{job.base}</code></b><br/>or<br/><b><code>#{job.name}</code></b></p> | <p>Returns the name of the applied Job Creation preset without any extension. Expands to <b>Untitled</b> if no Job Creation preset was used.</p>                                                                                                                                                                                                                                                                   |
| <p><b><code>#{job.nr}</code></b></p>                                              | <p>An automatic sequence number belonging to the current output file. It is automatically incremented for each new output file that gets created. Note, that multiple output files are created, for example, when output separation has been selected for output creation.</p> <p>It is possible to format the number using a pattern and locale. See "Formatting date and number values " on page 969, below.</p> |
| <p><b><code>#{job.ext}</code></b></p>                                             | <p>The extension that corresponds to the chosen output format, for example <b>pdf</b> or <b>ps</b>.<br/>Note that <code>#{job.ext}</code> does not include a leading dot.</p>                                                                                                                                                                                                                                      |

## Content variables

The variables listed below can be used in text, barcodes, and OMR and Image data in the "Additional Content" on page 799 page, and in the Conditional field when selectively adding inserts in the "Inserter Options" on page 865.

If the output is grouped and separated, Content variables on the separation level and above are **also available as File name variables**. For example, if the output is grouped on the job segment and document set level, and is to be separated on the Document Set level, the **set** and **segment** variables can also be used in the Job Output Mask field.

### Warning

Use `count` variables with caution. They entail higher memory usage in Weaver (the engine that creates Print output). When, for example, `segment.count.pages` is used in additional text, the Weaver engine has to buffer all pages until it knows how many pages the segment counts in order to include that total in the additional text on each page. With big jobs this could easily lead to Out of Memory errors.

## System

|                                    |                                                                                                                                           |
|------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| <b><code>#{system.time}</code></b> | Displays the current system data and/or time. Can be formatted using the "Formatting date and number values " on page 969, as seen below. |
|------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|

## Page

|                                                                                     |                                                                   |
|-------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <b><code>#{page.back}</code></b>                                                    | True when the page is on the back of the sheet.                   |
| <b><code>#{page.front}</code></b>                                                   | Boolean indicating whether the page is on the front of the sheet. |
| <b><code>#{page.height}</code></b>                                                  | The page's height (in points).                                    |
| <b><code>#{page.landscape}</code></b>                                               | True when the page's orientation is landscape.                    |
| <b><code>#{page.portrait}</code></b>                                                | True when the page's orientation is portrait.                     |
| <b><code>#{page.nr}</code> or<br/><b><code>#{page.sequence.document}</code></b></b> | Page number in the document.                                      |
| <b><code>#{page.sequence.sheet}</code></b>                                          | Page number on the sheet.                                         |
| <b><code>#{page.sequence.set}</code></b>                                            | Page number within the document set.                              |
| <b><code>#{page.sequence.segment}</code></b>                                        | Page number within the job segment.                               |
| <b><code>#{page.sequence.job}</code></b>                                            | Page number within the job.                                       |
| <b><code>#{page.width}</code></b>                                                   | The page's width (in points).                                     |

## Sheet

|                                          |                                            |
|------------------------------------------|--------------------------------------------|
| <b><code>#{sheet.count.pages}</code></b> | Total number of pages on the sheet.        |
| <b><code>#{sheet.duplex}</code></b>      | True when printing on the sheet is duplex. |

|                                                                                      |                                                                                                                                                       |
|--------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><code>#{sheet.height}</code></b>                                                  | The sheet's height (in points).                                                                                                                       |
| <b><code>#{sheet.pageDevice}</code></b>                                              | Array specifying the sheet's Type (e.g. Plain), Weight (e.g. 83), Color (e.g. yellow), Width (in points), Height (in points) and Tumble (e.g. false). |
| <b><code>#{sheet.nr}</code></b> or<br><b><code>#{sheet.sequence.document}</code></b> | Sheet number within the document.                                                                                                                     |
| <b><code>#{sheet.sequence.set}</code></b>                                            | Sheet number within the document set.                                                                                                                 |
| <b><code>#{sheet.sequence.segment}</code></b>                                        | Sheet number within the job segment.                                                                                                                  |
| <b><code>#{sheet.sequence.job}</code></b>                                            | Sheet number within the job.                                                                                                                          |
| <b><code>#{sheet.simplex}</code></b>                                                 | True when printing on the sheet is simplex (one-sided).                                                                                               |
| <b><code>#{sheet.width}</code></b>                                                   | The sheet's width (in points).                                                                                                                        |

## Document

|                                                                                                                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><code>#{document.metadata.<br/>propertyname}</code></b> or<br><b><code>#{document.metadata<br/>['propertyname']}</code></b> | <p>Value of a meta data property of the document.</p> <p>The <i>propertyname</i> must have been defined as a Tag Name on the Document Tags tab of the "Metadata options" on page 849 page in the <i>Advanced Print Wizard</i> or <i>Job Creation preset</i>.</p> <p>Note: this variable is only available if Separation based on Document has been selected on the "Separation options" on page 856 page in the <i>Advanced Print Wizard</i> or <i>Output Creation preset</i>.</p> |
| <b><code>#{document.count.pages}</code></b>                                                                                    | Total number of pages in the document.                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b><code>#{document.count.sheets}</code></b>                                                                                   | Total number of sheets in the document.                                                                                                                                                                                                                                                                                                                                                                                                                                            |

|                                                             |                                          |
|-------------------------------------------------------------|------------------------------------------|
| <b>`\${document.nr}` or<br/>`\${document.sequence.set}`</b> | Document number within the document set. |
| <b>`\${document.sequence.segment}`</b>                      | Document number within the job segment.  |
| <b>`\${document.sequence.job}`</b>                          | Document number within the job.          |

## Set

|                                                                                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|--------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>`\${set.metadata.<br/>propertyname}` or<br/>`\${set.metadata<br/>['propertyname']}`</b> | <p>Value of a meta data property of the Document Set.</p> <p>The <i>propertyname</i> must have been defined as a Tag Name on the Document Set Tags tab of the "Metadata options" on page 849 page in the <i>Advanced Print Wizard</i> or <i>Job Creation preset</i>.</p> <p>Note: this variable is only available if Separation based on Document Set has been selected on the "Separation options" on page 856 page in the <i>Advanced Print Wizard</i> or <i>Output Creation preset</i>.</p> |
| <b>`\${set.count.pages}`</b>                                                               | Total number of pages in the document set.                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>`\${set.count.sheets}`</b>                                                              | Total number of sheets in the document set.                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>`\${set.count.documents}`</b>                                                           | Total number of documents in the document set.                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>`\${set.nr}` or<br/>`\${set.sequence.segment}`</b>                                      | Document set number within the job segment.                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>`\${set.sequence.job}`</b>                                                              | Document set number within the job.                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

## Segment

|                                                                             |                                                                                                                         |
|-----------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------|
| <b>`\${segment.metadata.<br/>propertyname}` or<br/>`\${segment.metadata</b> | <p>Value of a meta data property of the job segment.</p> <p>The <i>propertyname</i> must have been defined as a Tag</p> |
|-----------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------|

|                                                                       |                                                                                                                                                                                                                                                                                                                                                                                                   |
|-----------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>[{propertyname}]</code>                                         | Name on the Job Segement Tags tab of the "Metadata options" on page 849 page in the <i>Advanced Print Wizard</i> or <i>Job Creation preset</i> .<br><br>Note: this is only available if Separation based on Job Segment or <i>Split At Exactly n Sheets</i> has been selected on the "Separation options" on page 856 page in the <i>Advanced Print Wizard</i> or <i>Output Creation preset</i> . |
| <code>#{segment.count.pages}</code>                                   | Total number of pages in the job segment.                                                                                                                                                                                                                                                                                                                                                         |
| <code>#{segment.count.sheets}</code>                                  | Total number of sheets in the job segment.                                                                                                                                                                                                                                                                                                                                                        |
| <code>#{segment.count.documents}</code>                               | Total number of documents in the job segment.                                                                                                                                                                                                                                                                                                                                                     |
| <code>#{segment.count.sets}</code>                                    | Total number of document sets in the job segment.                                                                                                                                                                                                                                                                                                                                                 |
| <code>#{segment.nr}</code> or<br><code>#{segment.sequence.job}</code> | Job segment number within the job.                                                                                                                                                                                                                                                                                                                                                                |

### Formatting date and number values

Date and number values can be formatted using an optional pattern and/or locale.

| Form                                      | Description                      | Example                                         | Result                            |
|-------------------------------------------|----------------------------------|-------------------------------------------------|-----------------------------------|
| <code>#{expression}</code>                | Do not format.                   | <code>#{system.time}</code>                     | July 4,<br>2009<br>12:30:55<br>PM |
| <code>#{expression,pattern}</code>        | Apply pattern with system locale | <code>#{system.time, yyyyMMdd-HH:mm:ss}</code>  | 20090704-<br>12:30:55             |
| <code>#{expression,pattern,locale}</code> | Apply pattern with the specified | <code>#{system.time, "dd MMMM yyyy", fr}</code> | 19<br>décembre                    |

|                                     |                                                          |                                       |                                    |
|-------------------------------------|----------------------------------------------------------|---------------------------------------|------------------------------------|
| }                                   | country locale                                           |                                       | 2017                               |
| <code>\${expression,,locale}</code> | Apply a default format with the specified country locale | <code>\${system.time,,fr}</code><br>> | 19<br>décembre<br>2017<br>12:30:55 |

It is possible to enclose the values of the pattern and locale in single or double quotes. This is required for including whitespace in a pattern, or when the `${expression}` would otherwise be ambiguous.

At run-time, the output engine determines the type of the value yielded by the expression. If this is a number, a number pattern is expected. For date/time-like types, a date pattern is expected. When no pattern is specified, some default format is applied. For other types, it is not possible to specify a pattern or locale.

## Generating Fax output

It is possible to generate Fax output from PlanetPress Connect through the use of PDF/VT output. Here are the details on how to implement such a process.

### Required Components

The following components are required in order to output to Fax:

- A PlanetPress Image license which includes PlanetPress Fax.
- A Job Preset adding the appropriate metadata fields
- An Output preset generating a PDF/VT file.
- A PlanetPress Workflow process outputting to the PlanetPress Fax task.

### Job Preset Configuration

The following metadata fields must be added to the [Metadata Options](#) page:

- **FaxNumber:** The phone number where the fax will be sent. Often part of the data.
- **FaxInfo:** The description of the fax in both the PlanetPressFax dialog box and the fax log file.

## Output Preset Configuration

The following settings must be used in the Output Preset:

- In the [Print Options](#), a PDF type should be selected, such as Generic PDF.
- In the [PDF Options](#), the PDF Type should be set to PDF/VT

## PlanetPress Workflow Process

The following Workflow will produce Fax output:

- The four regular Connect tasks to generate print output:
  - **Execute Data Mapping**
  - **Create Print Content**
  - **Create Job** using the above *Job Preset*
  - **Create Output** using the above *Output Preset*. The task's **Output Management** must be set to be *Through Workflow*.
- The **PlanetPress Fax** connector task set to Passthrough (the first "Document" on the list).

## Generating Tags for Image Output

It is possible, even easy, to generate specific tags and indexes for PlanetPress Image. This can be used to send email, archive with Search or output to image formats.

### Required Components

The following components are required in order to output to Image:

- A PlanetPress Imaging license.
- A Job Preset adding the appropriate metadata fields
- An Output preset generating a PDF/VT file.
- A PlanetPress Workflow process outputting to the PlanetPress Image task.

### Job Preset Configuration

For email output, the following metadata fields must be added to the [Metadata Options](#) page:

- **ImageSendTo:** Add to have PlanetPress Image use the specified field as the E-mail address to which to send the PDF file.
- **ImageSendCc:** Add to have PlanetPress Image use the specified field as the E-mail address to which to send the PDF file as a carbon copy (CC).
- **ImageSendBcc:** Add to have PlanetPress Image use the specified field as the E-mail address to which to send the PDF file as a blind carbon copy (BCC).
- **ImageSubject:** Add to have PlanetPress Image use the specified field as the subject of the email that is sent.
- **ImageBody:** Add to have PlanetPress Image use the specified field as the body of the email that is sent.

Note that the PDF file generated by the Print context is sent as an attachment to the email sent using the information above.

### PlanetPress Search Indexing

For PlanetPress Search indexing, you can add your own custom fields. Each field that is not included in the above or in [Generating Fax output](#) is added as an index for PlanetPress Search. For example you could add CustomerID and this would appear as the CustomerID index in Search. Yes, it's that easy!

### Output Preset Configuration

The following settings must be used in the Output Preset:

- In the [Print Options](#), a PDF type should be selected, such as Generic PDF.
- In the [PDF Options](#), the PDF Type should be set to PDF/VT

### PlanetPress Workflow Process

The following Workflow will produce Image output:

- The four regular Connect tasks to generate print output:
  - **Execute Data Mapping**
  - **Create Print Content**
  - **Create Job** using the above *Job Preset*



- **Create Output** using the above *Output Preset*. The task's **Output Management** must be set to be *Through Workflow*.
- The **PlanetPress Image** connector task set to Passthrough (the first "Document" on the list). If sending Email, choose the "Send Email" option of the Output group. Otherwise, choose Archive Output, ensure the output type is PDF, and optionally fill in the PlanetPress Search Database tab appropriately.

## Generating Email output

The Email context outputs HTML email with embedded formatting to an email client through the use of an email server. The HTML generated by this context is meant to be compatible with as many clients and as many devices as possible.

Email Output can be generated in two different ways: from the Designer or via Workflow. In both cases, email is sent in a single batch for the whole record set.

To test a template, you can send a test email first.

Output, generated from an Email template, can have the following attachments:

- The contents of the Print context, in the form of a single PDF attachment.
- The output of the Web context, as an integral HTML file.
- Other files, an image or a PDF leaflet for example.

Attaching the Print context and/or the Web context is one of the options in the Send (Test) Email dialog.

To learn how to attach other files, see "Email attachments" on page 379.

### Before generating Email output

- Decide on the use of an Email Service Provider; see "Using an ESP with PlanetPress Connect" on page 976.
- Make sure that a data set is loaded, that any necessary files, such as images and attachments, are in place, and that the correct settings are selected (see below).
- You may want to **rasterize** certain elements, such as business graphics. Rasterizing converts the element to a JPG or PNG image. This is very useful to support as many

clients as possible. For example, some email clients may not support SVG, so converting a resource to JPG instead would ensure that most email clients would actually see the output.

To rasterize an element, right-click it and select **Rasterize options**. For a JPG image you can set the quality of the resulting image in a percentage.

## Email output settings in the Email context and sections

The following settings for the Email context and Email sections have an impact on how the actual emails are sent.

- An Email To Script must be available in the template and refer to a valid email address; see "Email header settings" on page 373. If any record does not have a valid email, this record is skipped automatically when generating email output.

### Note

When you send a test email, the Email To Script will not be used; instead, the email will be sent to the address that you specify in the Send Test Email dialog.

- The sender(s), recipient(s) and the subject can be set using Script Wizards; see "Email header settings" on page 373.
- Default SMTP settings can be set in the preferences; see "Email header settings" on page 373.
- If there are multiple Email sections, only one of them can be merged with each record. Make sure that the correct section has been set as the default; see "Setting a default Email template for output" on page 373.  
To dynamically select a section for output, use a Control Script; see "Control Scripts" on page 645.
- PDF attachments can be compressed to make the files smaller; see "Compressing PDF attachments" on page 369.

## Generating Email output from Connect Designer

To generate Email output from the Designer:

1. Open a template with an Email context.
2. Load a data file or database compatible with this template, or open a Data Mapping Configuration. See "Loading data" on page 594.  
If you have an open Data Mapping Configuration and open another data file, the current Data Mapping Configuration will try to retrieve data from the file or database using its own Data Model and extraction logic.

### Note

When generating output with just an open Data Mapping Configuration, the template is merged with the complete sample data file that is part of the Data Mapping Configuration. The output is **not** limited to the number of records shown in the Data Model pane (which is one of the settings in the DataMapper).

3. On the **File** menu, click **Send Email** or **Send Test Email**. In the dialog that appears you can, among other things, attach the Print context or the Web context to the email. See "Send (Test) Email" on page 723 for a description of all the options. Finally, click OK.

### Note

#### About testing emails

When you send a test email, the Email To Script will not be used; instead, the email will be sent to the address that you specify in the Send Test Email dialog. If you have a Litmus account, you can enter your Litmus test address. To make the test address appear by default, you can set the default test address in the Email Preferences: select **Window > Preferences**, click the arrow next to **Email**, click **General** and type the test address next to **Email Test address**.

For a description of how to test your email for different email clients, see this how-to: [Test your emails with Litmus](#). For more information on Litmus, please see <http://litmus.com/>

### Tip

For a detailed description of how to use Mandrill with Connect to send and track emails, see the following how-to: [Using Mandrill](#).

## Generating Email output from Workflow

1. Open a template with an Email context.
2. Send the template to PlanetPress Workflow; see "Sending files to Workflow" on page 309.
3. Create a process in PlanetPress Workflow containing at least the following steps:
  - Any input that will capture a job file that is compatible with the data mapping configuration that is used.
  - An Execute Data Mapping task to generate a valid record set (see Workflow Help: [Execute DataMapping Task](#)).
  - A Create Email Content task with the appropriate settings (see Workflow Help: [Create Email Content](#)).

## Using an ESP with PlanetPress Connect

An email service provider (ESP) is a company that offers email marketing or bulk email services.

This topic explains why and how to use an ESP with PlanetPress Connect

### Reasons to use an ESP

These are a number of reasons why you would need an ESP:

- ESPs ensure a high deliverability, as most ESPs are whitelisted or approved by ISPs (Internet Service Providers) as legitimate email delivery service. So they help you to avoid having mail detected as spam.
- ESPs provide comprehensive tracking options to measure open rates and they log which links were clicked and by who. Typically this information is available via an online dashboard.

- Most ESPs provide Bounce Management options. They will stop sending messages to addresses that return a hard bounce and retry for soft bounces before removing that address.
- ESPs can handle unsubscribes and prevent accidental sends in the future.

## Choosing an ESP

The first thing to do to use an ESP with PlanetPress Connect is to choose an ESP and create an account.

Mandrillapp.com, a popular ESP, used to have a free account but now requires a paid MailChimp account. Luckily there are plenty of alternatives that provide free accounts (often capped to a max number of emails per month and sometimes having throttled output).

PlanetPress Connect has been tested with: Mandrillap.com, SendGrid (easy user management), MailGun (nearly instant statistics) and MailJet (shows best performance on the free account).

## Adding an SMTP Preset for an ESP

After creating an account, add a SMTP settings preset in PlanetPress Connect for the chosen ESP, via the Preferences dialog of the Designer (see "Email SMTP settings" on page 375).

Make sure **Use authentication** is checked, and put in your SMTP Username in the box below.

### Note

Presets for different ESPs are already available in the list of default presets.

## Sending an email with an ESP

To send an email or test email with the use of an ESP, start generating the email as usual (see "Generating Email output" on page 973). In the Send (Test) Email dialog, pay attention to the following settings:

- In the **Outgoing mail settings** area, select the preset for your ESP in the Presets drop-down.
- In the **Password** box, type the password provided by the ESP.

### Note

The ESP might also have a test function you can use. Check the options of your ESP.

### Tip

For a detailed description of how to use Mandrill to send and track emails, see the following how-to: [Using Mandrill](#).

## Adding custom ESP handling instructions

Most ESPs allow you to provide custom handling instructions as part of the email message, via custom headers. Typically these include instructions to enable open rate tracking, click through rate tracking and assign tags/categories to messages. Assigning a tag/category allows you to view statistics per email type in the dashboard of the ESP. Note that each ESP has its own notation and instructions.

In a Connect template, adding these custom headers is handled through a Control Script (see "Control Scripts" on page 645, "Control Script API" on page 930 and "section" on page 936). The following samples show how to assign a tag or category to a message (e.g. 'invoice', 'confirmation', 'newsletter-jan-2017') for various ESPs.

### SendGrid

Dashboard: <https://app.sendgrid.com/>

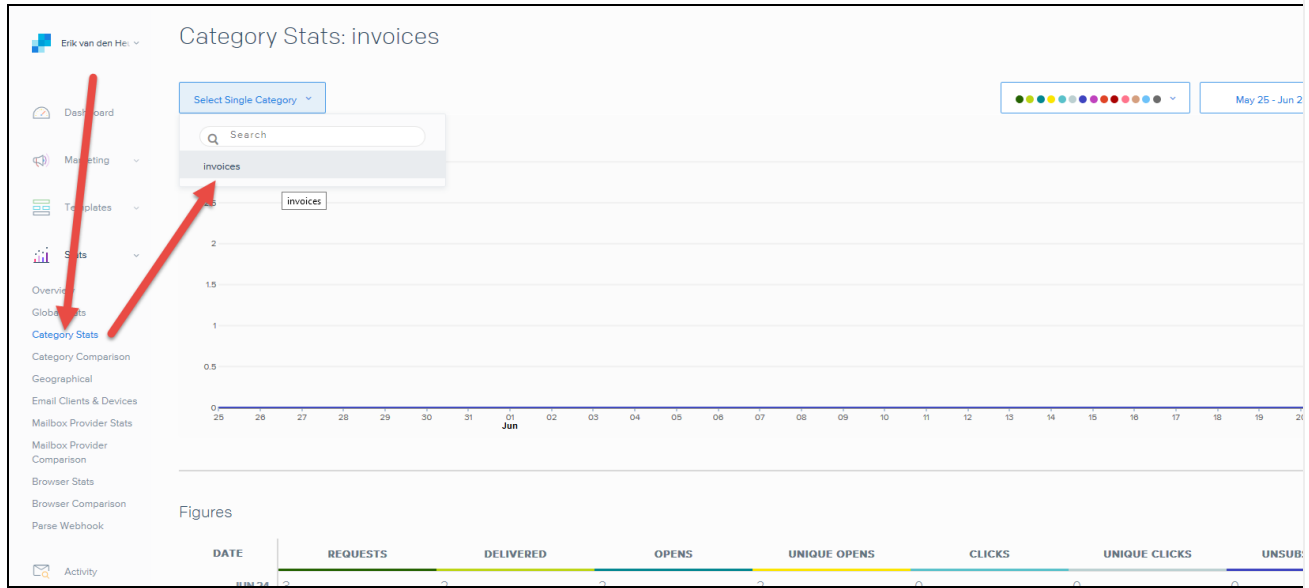
Documentation: [https://sendgrid.com/docs/API\\_Reference/SMTP\\_API/using\\_the\\_smtp\\_api.html](https://sendgrid.com/docs/API_Reference/SMTP_API/using_the_smtp_api.html)

Sample Control Script to assign a category:

```
var headerObj = {
 "category": ["invoices"]
};
merge.context.sections["Content"].headers = {
 "X-SMTPAPI": JSON.stringify(headerObj)
};
```

## Note

Sendgrid strips out their mail headers. The results need to be verified via their Dashboards (e.g. the Stats section lets you verify the stats for specific categories). Alternatively one can use their Web API to retrieve stats in JSON format. To view the category stats, log in to Sendgrid and choose: Stats > Category Stats > your category name.



## MailGun

Dashboard: <https://mailgun.com/cp/stats>

Documentation: <https://documentation.mailgun.com/api-sending.html#sending>

Sample Control Script to assign a tag:

```
merge.context.sections["Content"].headers = {
 "X-Mailgun-Tag": "invoices"
};
```

## Note

The Mailgun tag allows you to view the stats per tag. Mailgun has a quick refresh and stats are

available almost instantly.

The screenshot displays the Mailgun dashboard interface. At the top, there is a table of email statistics:

|                  |   |   |   |   |   |   |   |   |
|------------------|---|---|---|---|---|---|---|---|
| Bounces          | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Spam reports     | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Unsubscribes     | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Incoming         | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Posts via routes | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Below the table, a note states: "\* Note: Counters are UTC-based." The main section is titled "Categorized By Tag" and includes a "Delete Selected" button, a search input field labeled "Enter tag here", and a "Search" button. A table below shows the distribution of tags:

|                                                 | Jun 18 | Jun 19 | Jun 20 | Jun 21 | Jun 22 | Jun 23 | Jun 24 | Total |
|-------------------------------------------------|--------|--------|--------|--------|--------|--------|--------|-------|
| <input type="checkbox"/> invoices-tag Delivered | 0      | 0      | 0      | 0      | 0      | 0      | 10     | 10    |

Navigation arrows are present below the table. A note indicates "Mailgun domain's tag limit is 4000." The bottom section is titled "Open & Click Event Settings For The Current Domain" and features two dropdown menus: "Tracking clicks: Yes" and "and opens: Yes". A note below explains: "If you enable this feature, Mailgun will scan your messages for links and rewrite them." A link is provided: "Looking for webhooks? We've moved them to a new Webhooks page."

## MailJet

Dashboard: <https://app.mailjet.com/dashboard>

Documentation: [https://app.mailjet.com/docs/emails\\_headers](https://app.mailjet.com/docs/emails_headers)

Sample Control Script to assign a campaign:

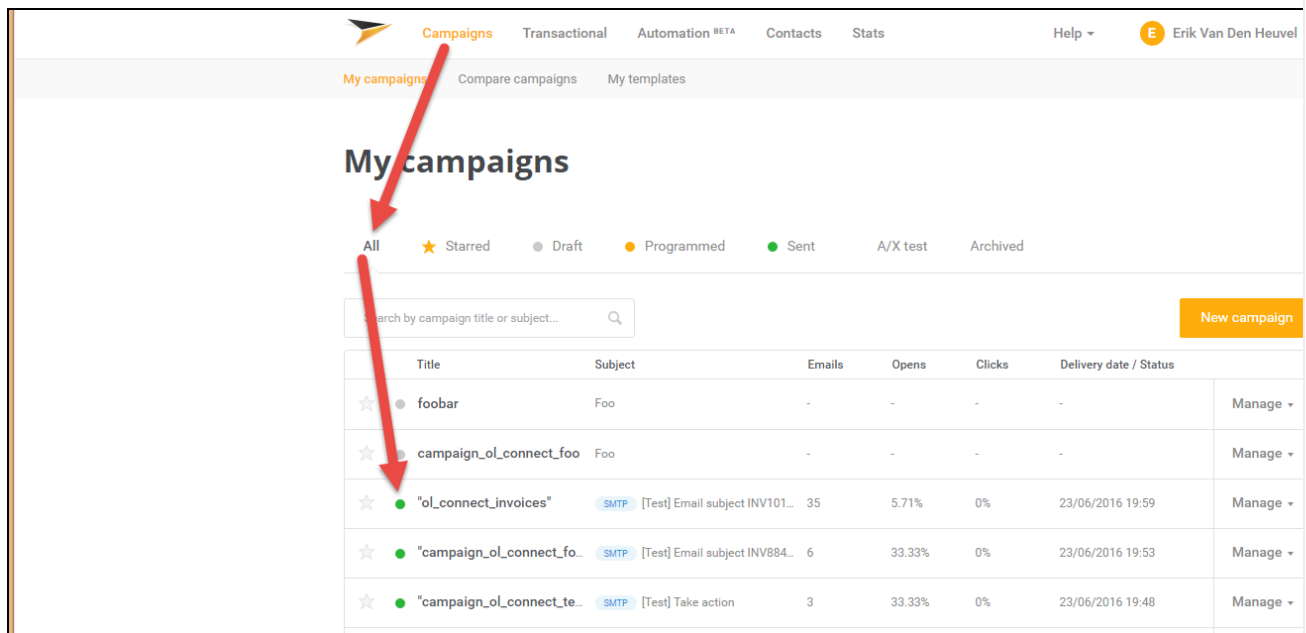
```
merge.context.sections["Content"].headers = {
 "X-Mailjet-Campaign": "invoices"
};
```

### Note

Mailjet strips out their own mailheaders like X-Mailjet-Campaign. The results can only be verified via the respective campaign stats page in the Mailjet dashboard. There is no need to pre-create the



campaign: adding it to the email header via a Control Script auto-generates the campaign. To view the campaign, login to Mailjet and choose: Campaigns > All.



The screenshot shows the Mailjet 'My campaigns' dashboard. At the top, there are navigation tabs: Campaigns, Transactional, Automation BETA, Contacts, and Stats. Below these are sub-tabs: My campaigns, Compare campaigns, and My templates. The main heading is 'My campaigns'. Below the heading are filter tabs: All (selected), Starred, Draft, Programmed, Sent, A/X test, and Archived. There is a search bar and a 'New campaign' button. The main content is a table with columns: Title, Subject, Emails, Opens, Clicks, and Delivery date / Status. The table contains five rows of campaign data.

| Title                        | Subject                               | Emails | Opens  | Clicks | Delivery date / Status |
|------------------------------|---------------------------------------|--------|--------|--------|------------------------|
| foobar                       | Foo                                   | -      | -      | -      | -                      |
| campaign_o_l_connect_foo     | Foo                                   | -      | -      | -      | -                      |
| "o_l_connect_invoices"       | [SMTP] [Test] Email subject INV101... | 35     | 5.71%  | 0%     | 23/06/2016 19:59       |
| "campaign_o_l_connect_fo_... | [SMTP] [Test] Email subject INV884... | 6      | 33.33% | 0%     | 23/06/2016 19:53       |
| "campaign_o_l_connect_te_... | [SMTP] [Test] Take action             | 3      | 33.33% | 0%     | 23/06/2016 19:48       |

## Generating Web output

The Web context outputs one HTML web page that contains the HTML text and all the resources necessary to display it. JavaScript files are added to the <head> in the generated HTML file. They are useful to add special features such as those offered by jQuery and its plugins, or MooTools. Style sheets are also added to the <head> and are used just as they would be used in a regular web page.

Web output can be generated in two different ways: it can be attached to an Email template when generating Email output, or it can be generated using Workflow.

Web output can be generated from the Designer when a data set is available. The data can be retrieved from a database or data file, or from a Data Mapping Configuration.

If you have an open Data Mapping Configuration and open another data file, the current Data Mapping Configuration will try to retrieve data from the file or database using its own Data Model and extraction logic.

### Note

When generating output with just an open Data Mapping Configuration, the template is merged with the complete sample data file that is part of the Data Mapping Configuration. The output is **not** limited to the number of records shown in the Data Model pane (which is one of the settings in the DataMapper).

### Before generating Web output

Before actually generating the Web output, you may want to **rasterize** certain elements, such as business graphics. Rasterizing converts the element to a JPG or PNG image. This is very useful to support as many clients as possible. For example, when a heading uses a font type that is not a Web font, converting the heading to JPG instead would ensure that the heading looks the same in all browsers.

To rasterize an element, right-click it and select **Rasterize options**. For a JPG image you can set the quality of the resulting image in a percentage.

## Attaching Web output to an Email template

To attach the Web context to Email output:

1. Open a template with a Web context and an Email context.
2. Check the output settings for both contexts; see "Email output settings in the Email context and sections" on page 974 and "Web output settings in the Web context and sections" on the next page.

### Note

When adding the Web context to an email, only the default Web section is generated and added to the email as an HTML file. To attach multiple Web sections as separate attachments, you need to create a Control Script that specifies **parts**; see "Control Scripts" on page 645 and "Control Script API" on page 930.

3. Load a data file or database compatible with this template. See "Loading data" on page 594.

4. On the **File** menu, click **Send Email** or **Send Test Email**. In the dialog that appears, check the option to attach the Web context to the email. See "Send (Test) Email" on page 723 for a description of all options.

#### Note

When you send a test email, the Email To Script will not be used; instead, the email will be sent to the address that you specify in the Send Test Email dialog.

5. Fill in the dialog and send the emails.

## Generating Web output from Workflow

1. Open a template with a Web context.
2. Send it to Workflow using the Package Files dialog; see "Sending files to Workflow" on page 309.
3. Create a process in Workflow containing at least the following steps:
  - Any input that will capture a job file that is compatible with the Data Mapping Configuration that is used. To capture incoming web requests, such as from a personalized URL (see "Personalized URL" on page 623), use a HTTP Server Input task (see Workflow Help: [HTTP Server Input](#)).
  - An Execute Data Mapping task to generate a valid record set (see Workflow Help: [Execute DataMapping Task](#)).
  - A Create Web Content task with the appropriate settings, to generate the HTML output (see Workflow Help: [Create Web Content](#)).

## Web output settings in the Web context and sections

There are a few settings for the Web context and Web sections that have an impact on the actual web page that is generated.

These settings are:

- Which Web section is the 'default'; see "Setting a default Web page for output" on page 391. When generating Web output, if there are multiple Web sections, only one of them can be merged with each record.

- The title, shortcut icon and meta tags appearing in the web page's header. See "Setting the title, meta data and a shortcut icon" on page 392

This chapter contains the current and previous versions of PlanetPress Connect 1.8 release notes.

[PlanetPress Connect 1.8 Release Notes.](#)

#### Previous releases:

- [PlanetPress Connect 1.7.1 Release Notes](#)
- [PlanetPress Connect 1.6 Release Notes](#)
- [PlanetPress Connect 1.5 Release Notes](#)
- [PlanetPress Connect 1.4 Release Notes](#)

## Overview

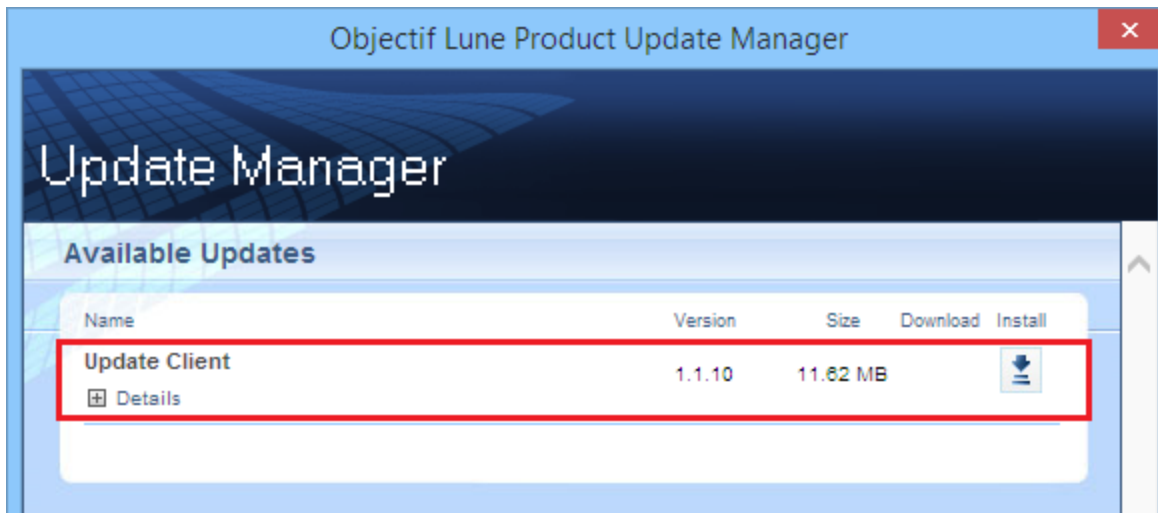
This document provides an overview of the new features and enhancements in PlanetPress Connect 1.8 and PlanetPress Workflow 8.8, as well as some important installation information.

### Upgrading from PlanetPress Connect 1.7

It is highly recommended that you update the **Objectif Lune Update Client** before upgrading PlanetPress Connect from version 1.7 to version 1.8.

If you do not update the Update Client, an unexpected error might occur whilst updating Connect. This error does *not* prevent the successful upgrade of Connect to 1.8, even though it appears as if it might have. To avoid potential confusion, we recommend that you first update the **Objectif Lune Update Client** before attempting to upgrade Connect from version 1.7 to version 1.8.

The Update Client will show that there is an update available for itself. Simply click on the download button in the dialog to install the new version of the Update Client. Note that it is no problem in running the update while the Client itself is still open. It will automatically update itself.



### Upgrading from PlanetPress Connect 1.1

In order to upgrade from Connect Version 1.1 to Version 1.8 via the Update Manager, it is necessary to install a newer version of the Objectif Lune Update Client. The next time you run your current Update Client it will show that there is an update available for itself. Simply click on the download button in the dialog to install the new version of the Update Client. Note that it is no problem in running the update while the Client itself is still open. It will automatically update itself.

Once you have done this, PlanetPress Connect 1.8 will become available for download.

From Connect Version 1.2 onwards, the newer version of the Update Client was included with the Connect installation by default.

### Installing PlanetPress Connect 1.8 and PlanetPress Workflow 8.8

- PlanetPress Connect is released as a 64 Bit version only (with the exception of the Workflow, Fax, Search and Imaging modules).
- Full details on installing and licensing PlanetPress Connect and PlanetPress Workflow can be found in the online help, which can be accessed from the software and the installer.
- Note that both PlanetPress Connect and PlanetPress Workflow come with 30 day trial license by default.

## **Updating stand-alone Workflow Messenger installations**

If Workflow Messenger was installed stand alone with no other Workflow components installed, the Update Client will be unable to find the Messenger component and thus it will not automatically update to the Workflow 8.8 version. To get around this, download and run the Workflow 8.8 installer manually.

## **Print Only Version**

A Print Only license is available with version 1.8 of PlanetPress Connect which allows legacy PlanetPress Suite 7 customers on OL Care to upgrade to Connect for a minimal fee. The license allows regular printing via the Print Wizard but runs Email and Web output in demo mode. For more information, please contact your local OL Customer Care or Sales team.

## **Templates Used in Workflow**

For improved performance we recommend re-saving Workflow templates set up in the previous versions to run with PlanetPress Connect 1.8\Workflow 8.8.

## **Reduced Memory Version**

It is possible to install PlanetPress Connect on a machine with a minimum of 2 GB of RAM. The PlanetPress Connect Designer will automatically detect whether it has been installed on a machine with less than 4 GB of RAM and default to only using one internal Weaver and one internal merge engine on that system. The Server will also run using internal engines.

**NOTE: This is *not* recommended for production.**

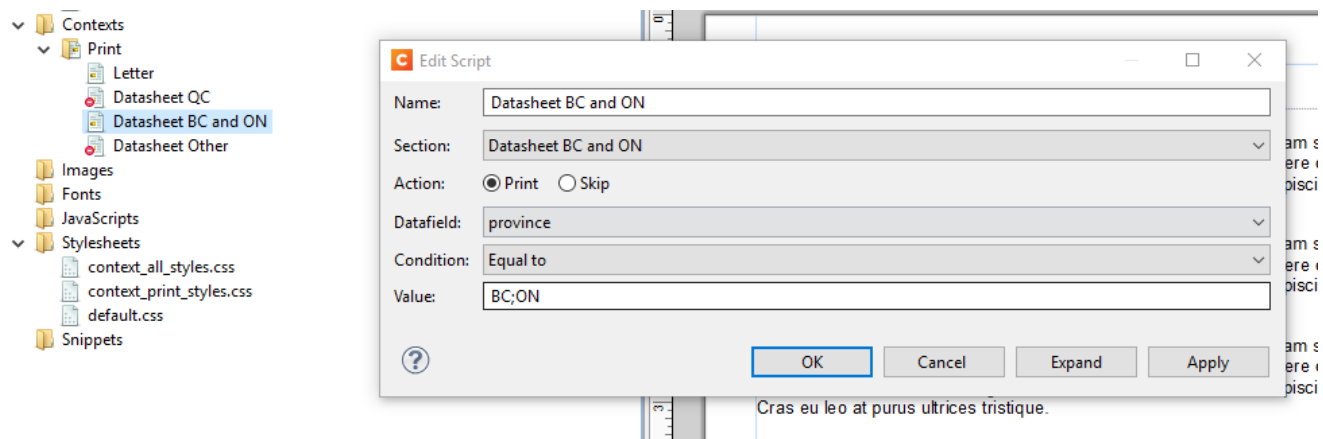
# Connect 1.8 General Enhancements and Fixes

## Native support for Microsoft Excel spreadsheet files

The Connect DataMapper can now handle Microsoft Excel files natively. The CSV data type has been enhanced to automatically recognize \*.XLS and \*.XLSX files and use them directly without any additional steps. Both the *CSV Wizard* and the *Add Data* options now allow you to pick these file types. You can even use different types of sample files regardless of their extension, as long as their formats match the data mapper configuration. (SHARED-58610)

## Conditional Print Sections Wizard

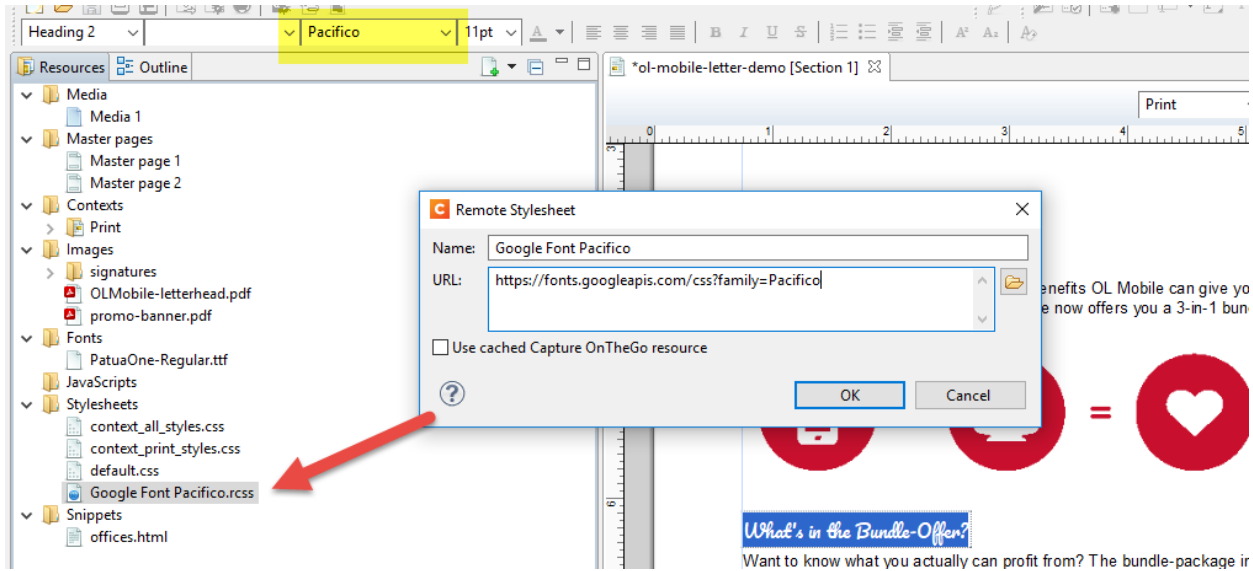
A wizard has been created to simplify the process of creating conditional print sections. Simply right mouse click a print section in the *Resource* panel and choose *Make Conditional...* The wizard allows you to enter a basic condition based on a data field value and either *skip* or *print* the section. When the specified condition is *true* the selected *Action* is applied (which can be Print or Skip) otherwise the opposite is in effect. The icon of the section in the *Resources* panel identifies if the section is printed or skipped. Use the *Expand* option to reveal the underlying *Control Script*. (SHARED-47661)



## Simplified Web Font support

Online font resources (such as [Google Fonts](#)) allow you to use their fonts for both commercial and non-commercial projects. Using these online fonts in your documents is a matter of linking to the Stylesheet file hosted by these services. Simply create a *Remote Stylesheet* entry and

paste the location of Stylesheet file in the *URL* field. This will make the font available to the application. It will be added to the Fonts menu and can be used in your custom CSS Stylesheets, via the `font-family` property. (SHARED-56637).



## Dynamically set Media Background Images

Support has been added for dynamically setting the path of media backgrounds at run time (aka Virtual Stationery). This is achieved via the Control Script API. The path can be set to an image in the *Images* folder but also to a file on disk (the `http://` and `https://` protocols are not currently supported). This greatly simplifies template management in situations where a design is shared between different brands. This technique can also be used to dynamically set the stationery image for the preview template in Connect Send environments. (SHARED-53522)

```
var myMedia = merge.template.media["Media 1"];
myMedia.stationery.front.enabled = true;
myMedia.stationery.front.url = "file:///C:/letterhead.pdf";
```

## Page Breaks inside Lists

Support has been added to allow the splitting of lists across pages. This includes *Widows* and *Orphans* control for ordered (`<ol>`) and unordered (`<ul>`) lists.

The *Orphans* CSS property specifies the minimum number of lines at the bottom of a page and the *Widows* property specifies the minimum number of lines after the page break. To prevent page breaks inside these elements simply add `page-break-inside: avoid;` to your stylesheets. A formatting dialog for these elements will be added in a future version. (SHARED-14092)




## Installer improvements

- Improved error capture, handling and messaging. (SHARED-40209)
- Significantly improved logging of Server Service installation. (SHARED-50796)

## Korean Language Support

PlanetPress Connect 1.8 is now available in Korean, in addition to the other languages already supported. Korean is not yet available in PlanetPress Workflow, however. (SHARED-40161)

## Context Sensitive Help

Context Sensitive Help has been added to PlanetPress Connect. Selecting Help (via  button, or **F1**) in dialog boxes or screens will now generally take you to the Help page most closely associated with the calling dialog box or screen. Context sensitivity will continue to be incrementally introduced and improved hereafter. (SHARED-45766)

## Database connection deadlocks resolved

We have added a timeout period to all Connect back-end database connectors, as well as increasing the amount of database connection threads to better match the capabilities of the hardware. This greatly reduces the chances of database deadlocks or bottlenecks when processing jobs through either Connect directly or through Workflow. (SHARED-57240/57252)

## Improved error handling of Merge Engine errors

In some rare circumstances XPCOM initializations would fail in Merge Engines. Thereafter the Merge Engine would continue to run but would be unable to process any further requests. Additional Merge engines would then be launched but the originals did not shut down, eventually leading to resource shortages and subsequent job failures. This issue has been addressed and XPCOM initialization errors in Merge Engines now cause the Merge engine to terminate and restart cleanly. (SHARED-57270)

## Anchored positioned boxes losing style attributes

Absolute positioned (Anchored) elements would lose some style attributes under certain circumstances. These issues would only occur when the absolute positioned element had multiple style attributes that ended with the text "*top*" or "*left*". Such as is the case with "*padding-top*" and "*top*". If both those attributes were set, then only one of the attributes would be retained. (SHARED-57361)

- Customers upgrading from 1.6.1 to 1.8 will not experience any issues with their templates.
- Customers upgrading from 1.7.1 or 1.7.2 to 1.8 will experience problems only if they have saved their templates within 1.7.1. or 1.7.2, and only if those templates contained absolute positioned objects with specific inline CSS styles that end with **top** or **left**, such as **padding-top**, **padding-left**, **border-top** etc.  
In that case those specific styles will be gone and they will either need to restore a backup from before 1.7.x of those templates or manually set the styles again in 1.8 and save the templates.

## Changes made to Output Speeds in Connect 1.8

A speed throttling issue was discovered that allowed some users to exceed license limitations. This issue has been corrected, and output speeds will now more accurately reflect license speeds.

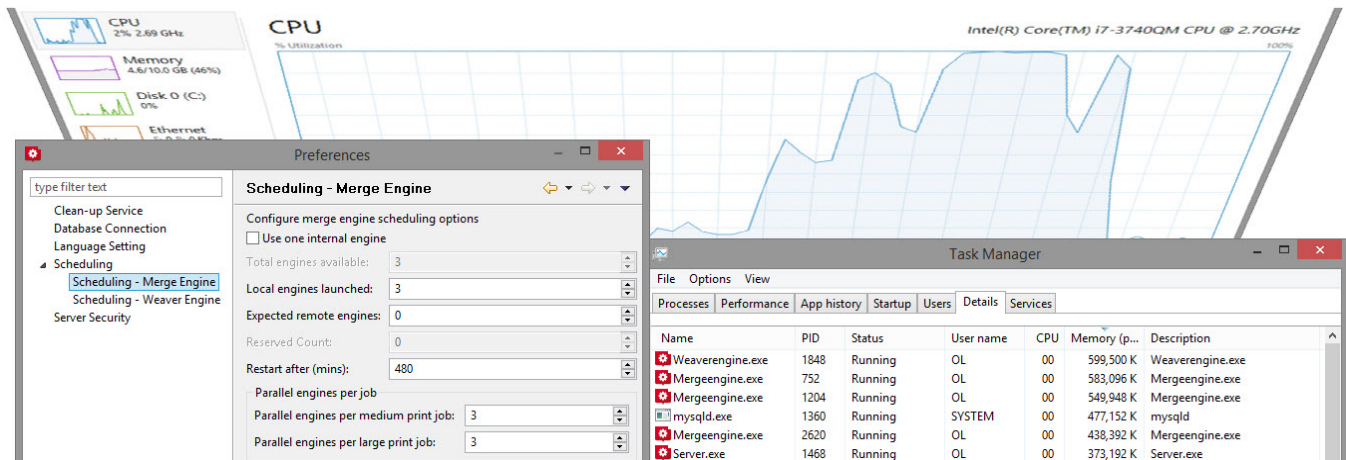
# Connect 1.8 Performance Related Enhancements and Fixes

## Faster Performance Tweaking in Server Configuration

When tweaking performance it can be hard to figure out the right settings for the number of Merge engines, Weaver engines, dividing speed units, etc. Having to restart the Connect Server to apply the changes every time a setting was changed also made tweaking performance harder than it should have been. So we've made some improvements in order that changing scheduling settings no longer requires a Server restart.

When you hit *Apply* or *OK*, the Connect Server will pick up your new settings. Engines are only stopped or started as needed, so the impact is minimal. Changes can even be made while jobs are running, without the jobs being interrupted.

This greatly reduces the time and effort required for optimizing performance. (SHARED-53222)



### Note

In the case that all engines are occupied, some changes will only take effect when the job finishes.

# Connect 1.8 Designer Enhancements and Fixes

## Automatically Fit Text to Container (Copy Fit)

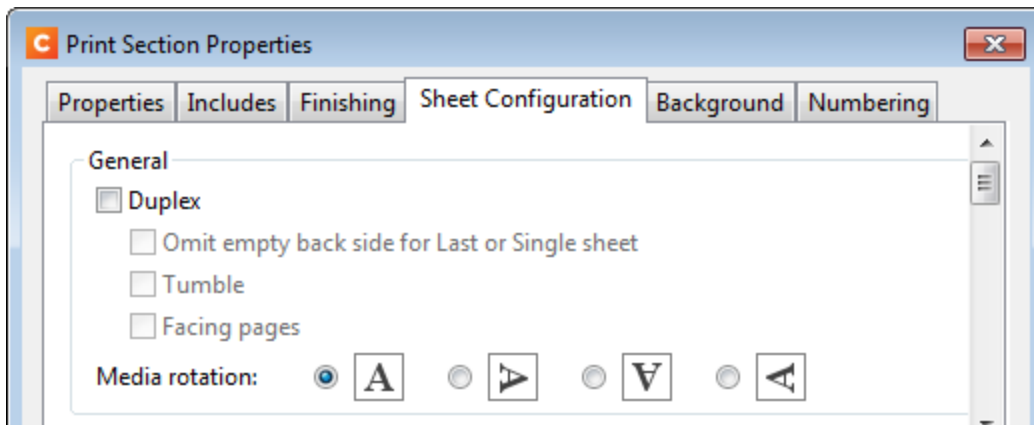
The Designer can now automatically scale text content to fit the boundaries of a box (inline or absolute positioned `<div>`). Scaling text to fit a container is a very popular feature when creating personalized post cards and the like.

The option is found in the *Content* tab of the *Box* properties dialog and can be set to scale all text or a specific element in that box by entering a CSS selector. (SHARED-37702)



## Rotate Print Sections individually

An option has been added to rotate individual *Print Section* orientations via the *Sheet Configuration* dialog (SHARED-46086)



## Snap Guidelines to Ruler

Dragging a guide whilst holding the Shift key will snap the guideline to the closest mark on the ruler. (SHARED-54465)

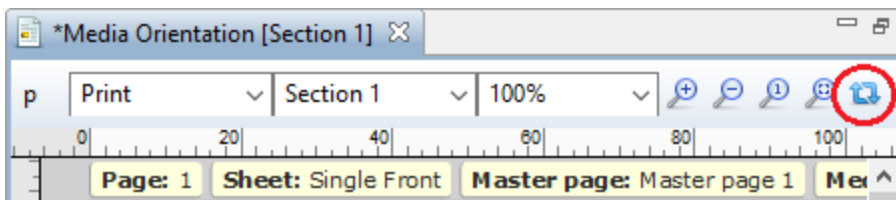
## Toggle Comments On/Off via Shortcut Keys

Toggle comments off or on in HTML, CSS and JavaScript editors via a keyboard combination. Use `Ctrl + /` to comment out a single line and `Ctrl + Shift + /` to comment out multiple lines. (SHARED-56440)

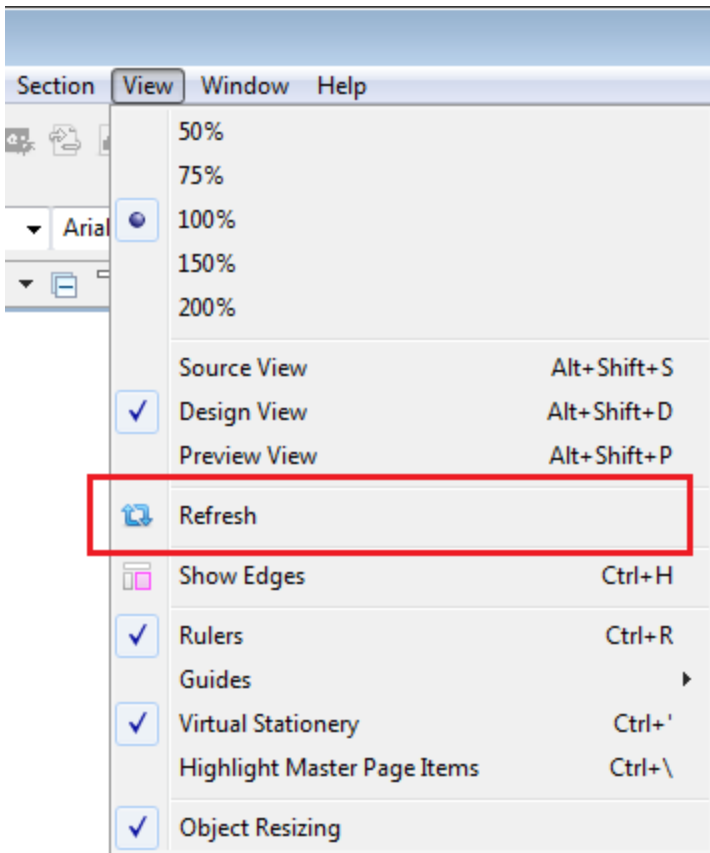
## Refresh View button added

You can now refresh the contents of both Design and Preview views via the new Refresh button or new Refresh selection in the Menu. (SHARED-55616)

Button :

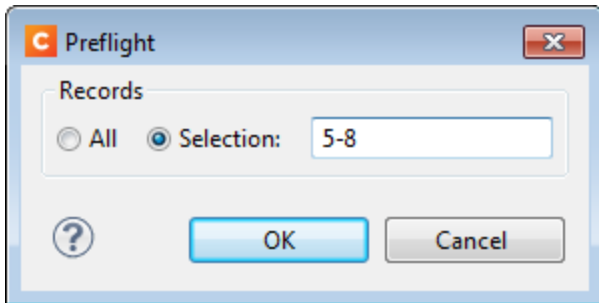


Menu selection :



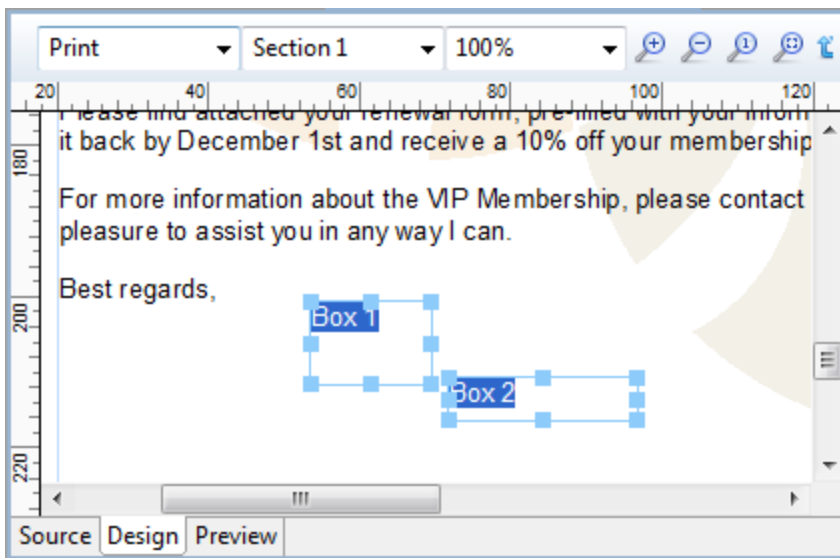
## Specify Page Range for Preflight

You can now optionally perform Preflights on a range of records. (SHARED-35076)



## Select and adjust multiple Box elements simultaneously

Multiple Box elements can now be selected at the same time.



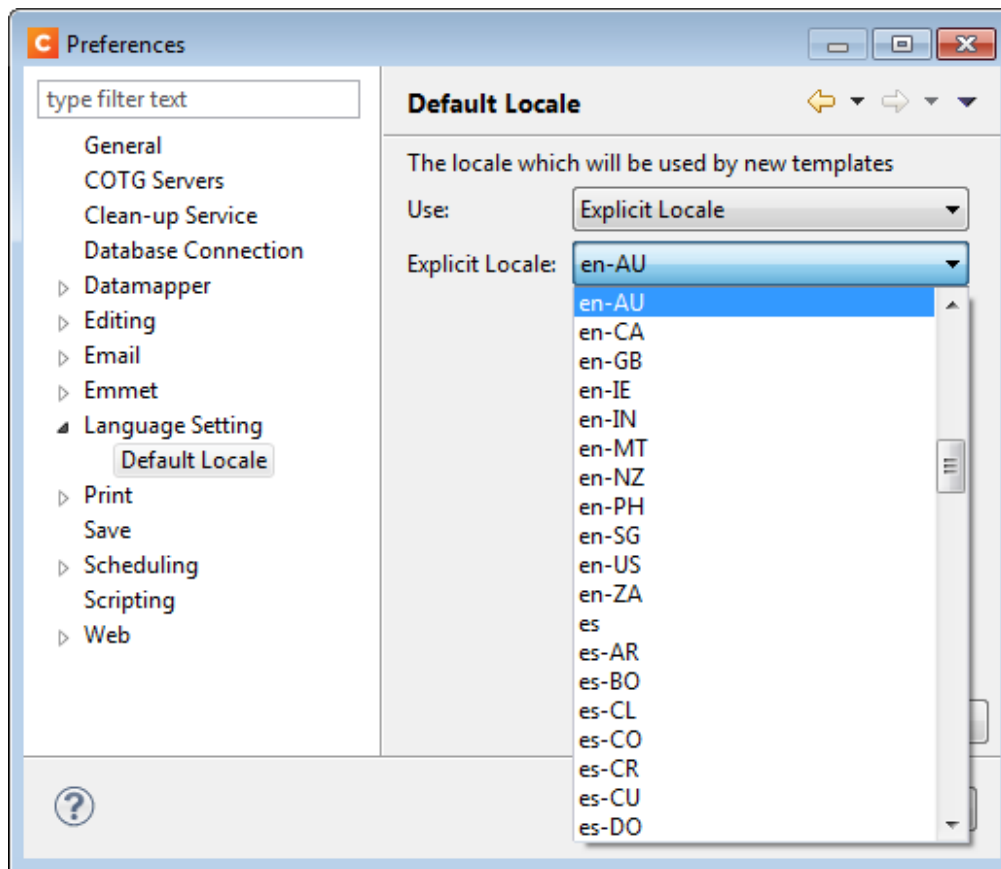
Once selected you can either move or resize the selected boxes as a group. You can move them either via the mouse, or by nudging them around a single pixel at a time with the arrow keys. When nudging, the boxes will not snap to guides. Otherwise, when moving or resizing multiple boxes, the box that was originally clicked (the reference box) will snap to guides if *Snap to Guides* is enabled. (SHARED-55636)

## Specify/change name of Email Attachments

Ability added to overwrite the file name for email attachments through scripting. (SHARED-57120)

## Set Template Locale

Previously Connect always assigned the *System Locale* to new templates. A new option has been added to the preferences to allow the selection of a specific locale. This selection will then apply to all new templates thereafter. It applies to Date and Time fields plus numeric and currency data fields. For example, a monetary data value in France (locale fr-FR) would apply the € (Euro) currency symbol and use the ',' (comma) as the decimal separator, whilst the same monetary data value in the US locale would apply the \$ (Dollar) symbol, and use the '.' (full stop) as the decimal separator. (SHARED-56791)



Locales can still be changed in individual templates by the **Edit > Locale ...** option.

## General Designer improvements

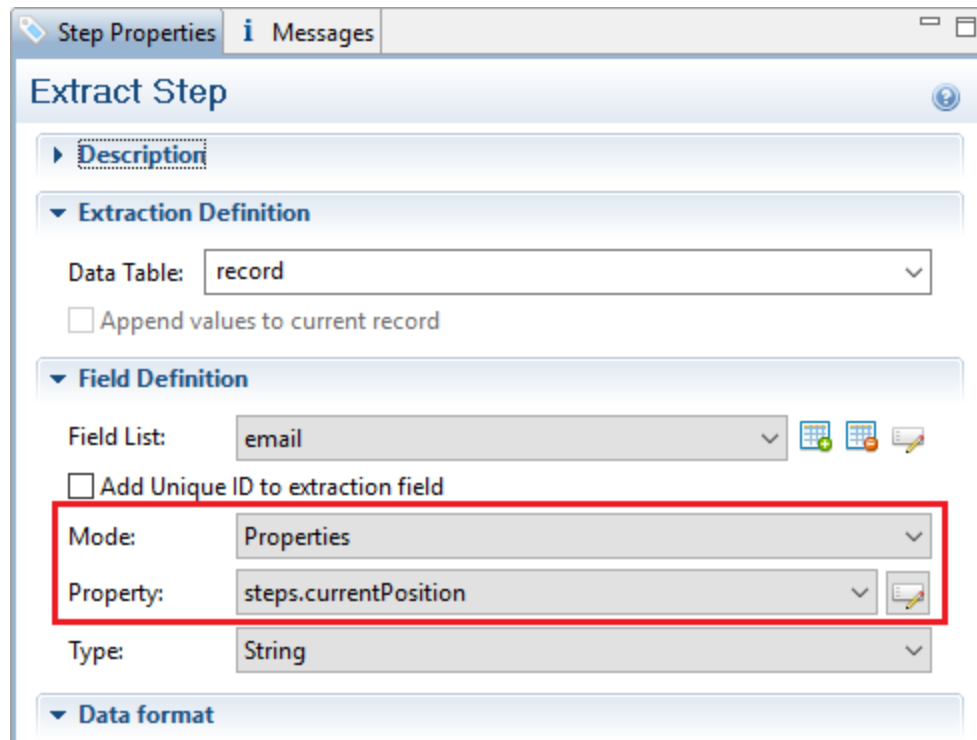
- **GS1 Datamatrix** barcode now supported. (SHARED-55999)
- Section, Media, and Master resources can now be duplicated by copy-pasting. (SHARED-52261)
- **Reopening a template** will put the focus on the section that was active when the template was closed. (SHARED-53199)
- **Keyboard shortcuts** to increase (Ctrl + Shift + >) or decrease (Ctrl + Shift + <) text size now work as expected. (SHARED-11660)
- "Problem" view renamed more accurately as "**Preflight Result**". (SHARED-56343)
- Added "**folding**" support for CSS editors and source editors for standalone HTML files. (SHARED-10717)
- Multiple values now allowed in the **Make Conditional** script wizard. (SHARED-57029)
- **Default colour** swatches can now be edited or overwritten. (SHARED-53670)
- Minor Designer interface inconsistencies fixed. (SHARED-54114)



# Connect 1.8 DataMapping Enhancements and Fixes

## Extracting Variables without using JavaScript

Variables are frequently used in data mapping configurations as counters or as a way to concatenate values before extracting the final result to a data model field. Until Connect 1.8, the only way to extract those variables would be to create a Javascript-mode field and to use the appropriate API syntax (e.g. `automation.jobinfo.jobinfo1`, or `sourceRecord.properties.MyVariable`). We've now made it much easier by providing a pick list of all available properties in the **Extract Step**. No JavaScript required!. (SHARED-54139)



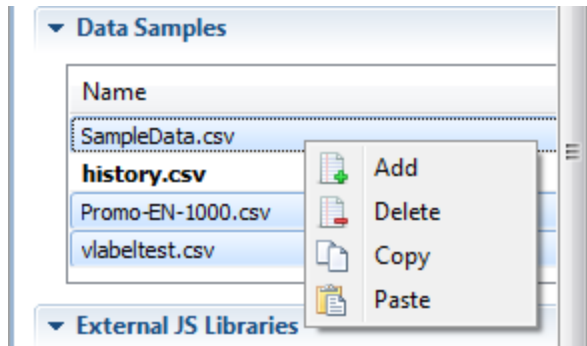
## "Extract all from here" feature added

In TXT mode, we've often had the request to be able to extract everything after a specific location, regardless of the record length. This would be useful, for instance, when extracting the body of an email which is never fixed-length. Until PlanetPress Connect 1.8, the customary way of proceeding would be to use a Loop step that stores all lines in a variable and an Extract step after the loop to store that variable into a field. Not the most elegant method! A new feature allows you to specify that a field should extract everything from the current location down to the end of the current record. No loops, no variables required. (SHARED-54137)

## Improved Management of Sample Data

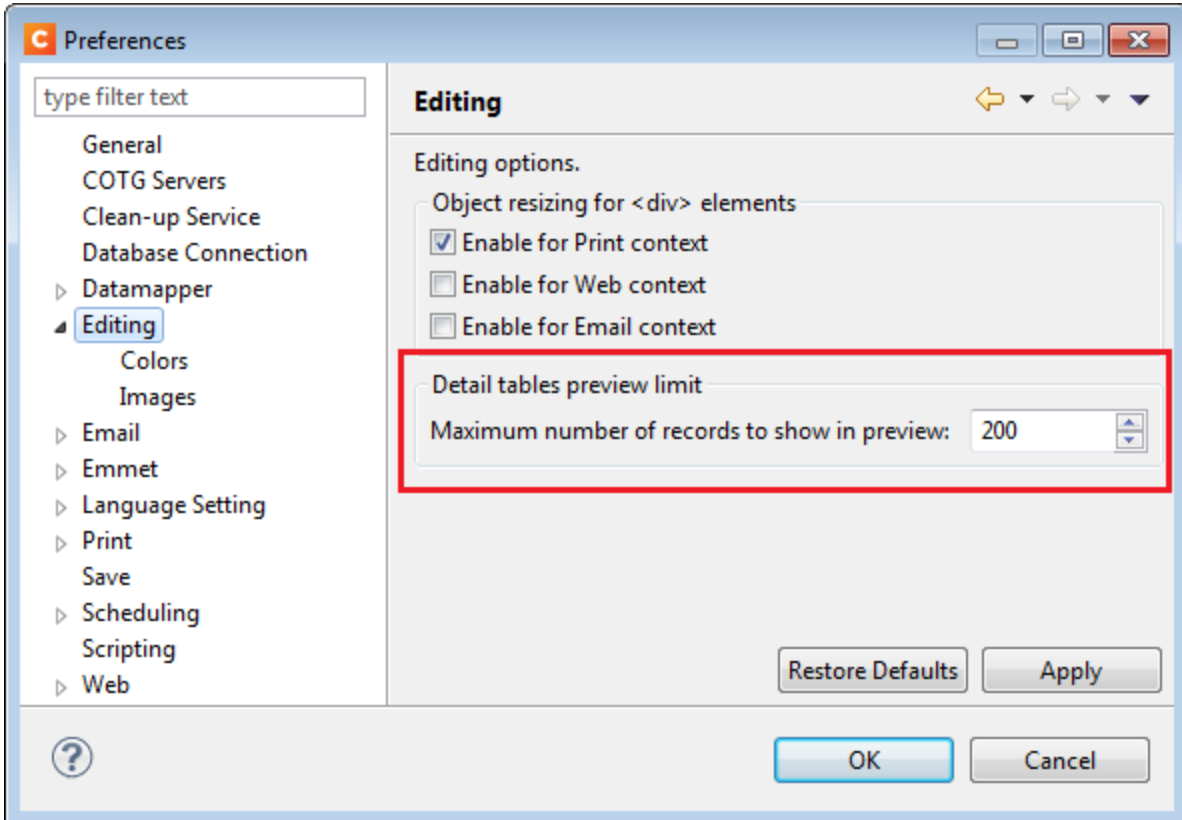
Support added for copy-and-pasting files from Windows Explorer to the Data Samples list, and vice-versa. This allows the quick exporting of those files without having to open the data mapping configuration. It also allows you to quickly add files to the configuration without having to use the Browse button.

In addition, you can now multi-select files in the Data Samples list and delete them at once, which saves you time when you want to remove all these test files from the configuration before sending it to production. (SHARED-43317/51800)



## Improvements made for Transactional Style Datafiles.

- The DataMapper and Output have both been enhanced to better cater for extremely large data records (where thousands of details might be associated with a single record). (SHARED-51691).
- The Designer has been updated to allow a limit to the maximum number of records to display in preview.



- Tooltip now displays a warning if detail table exceeds the preview limit. (SHARED-58212)

## Minor DataMapper Enhancements and Fixes

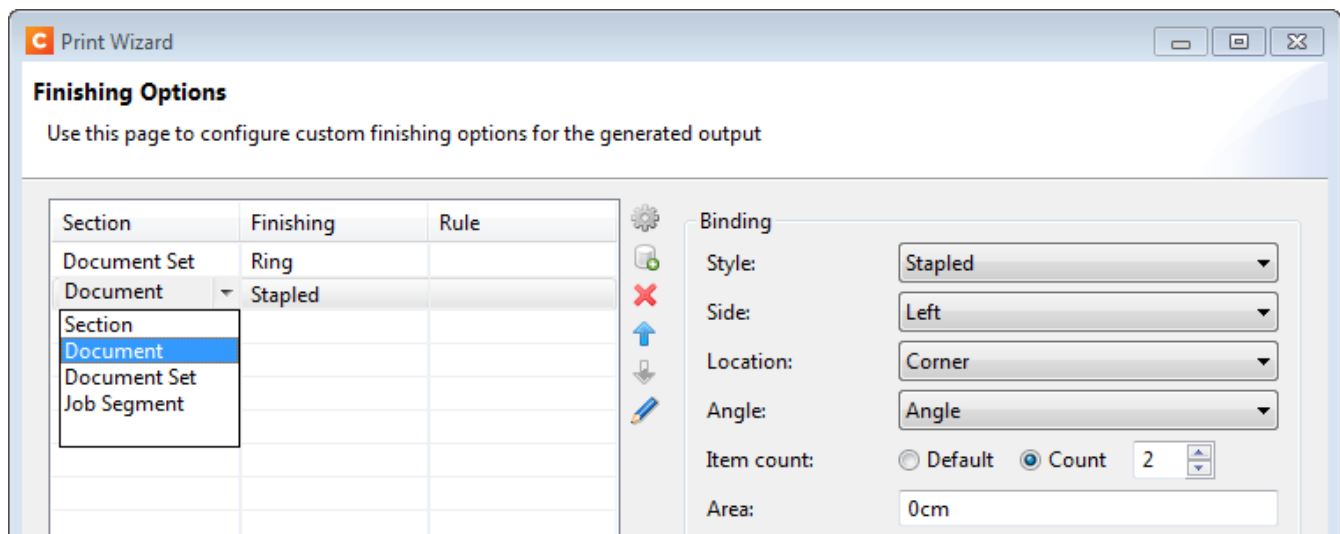
- Several other small improvements made to DataMapper interface. (SHARED-57347/33912)

# Connect 1.8 Output Enhancements and Fixes

## Dynamic Finishing

Print Output Finishing has been improved considerably, and is now much more powerful and flexible. In Templates, you could already set finishing for documents and sections. Job Creation would allow you to specify a different kind of Finishing for your documents and Templates. This has been extended to allow Finishing settings on all levels of Job Creation: **Document Sets**, **Job Segments** and **Jobs**. (SHARED-53277)

So now you can staple document sets and punch holes in your segments. You can also have multiple finishing settings at the same level.



The reason we call this feature **dynamic** finishing, is that it includes a brand new rules editor to allow you to choose when to apply a finishing setting:

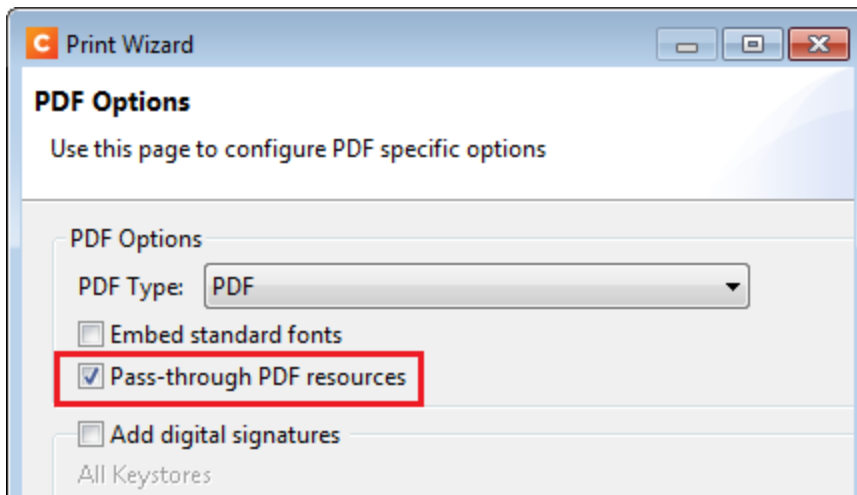
The screenshot displays a rules editor interface with the following components:

- Root Rule:** A dropdown menu set to "All of the following" with a green plus icon on the right.
- Group 1:** A dropdown menu set to "Not any of the following" with a red minus icon and a green plus icon on the right.
  - Rule 1:** "Field" dropdown, "abc Country" value, "is equal to" operator, empty value field, and "Case sensitive" checkbox.
  - Rule 2:** "Field" dropdown, "abc Country" value, "is equal to" operator, "CA" value, and "Case sensitive" checkbox.
- Group 2:** A dropdown menu set to "Any of the following" with a red minus icon and a green plus icon on the right.
  - Rule 3:** "Field" dropdown, "# InvTotal" value, "is greater than" operator, "0" value.
  - Rule 4:** "Field" dropdown, "Date" value, "is between" operator, "MonthBegin" value, "and" connector, and "DueDate" value.
- Rule 5:** "Size" dropdown, "Document Set" value, "has less than" operator, "5" value, and "Sheets" value.
- Rule 6:** "Position of" dropdown, "Document" value, "is first" operator, and "in Document Set" value.

## PDF Pass-through

Connect's output creation (Weaver engine) tries to write content the best way possible, depending on the chosen output format and optimization settings. However, there are cases when this might not be desired, such as when the graphics have already been optimized for the device and you do not want the software to change them.

It is now possible to instruct output creation to include PDF resources in the output file *as-is*. When used, it guarantees that the fidelity of PDF graphics used in a template is retained in the output. The resulting output will be less optimized, typically producing somewhat larger files. This option can also be useful when the output is showing unexpected results or to prevent rasterization of PDF output. Will this feature trigger visible differences in the output? No, in most cases not, but when printing highly optimized graphics, expect to see a slight difference in the printed output. This is only possible when PDF content is allowed in the output, meaning it can be used with PDF. For PDF output, this feature can be found in the Output Preset or Print Wizard page under **PDF Options**. (SHARED-56412)



Please note that when using a PDF from a Data Mapping in combination with Virtual Stationery, the PDF is not passed through when selecting this option in Connect 1.8. This is a known issue and will be addressed in a later release of OL Connect.

A second issue when using a PDF background via the Datamapper is that the resultant PDF output file *may* contain invalid font resources. Whilst the output can be viewed in Adobe Acrobat Reader without issue and will print correctly on many printers, it will prompt warnings in Adobe Acrobat Professional's Preflight report and it should not be used as input for Connect Data Mapping. We recommend testing the output on your specific printer(s) to best determine

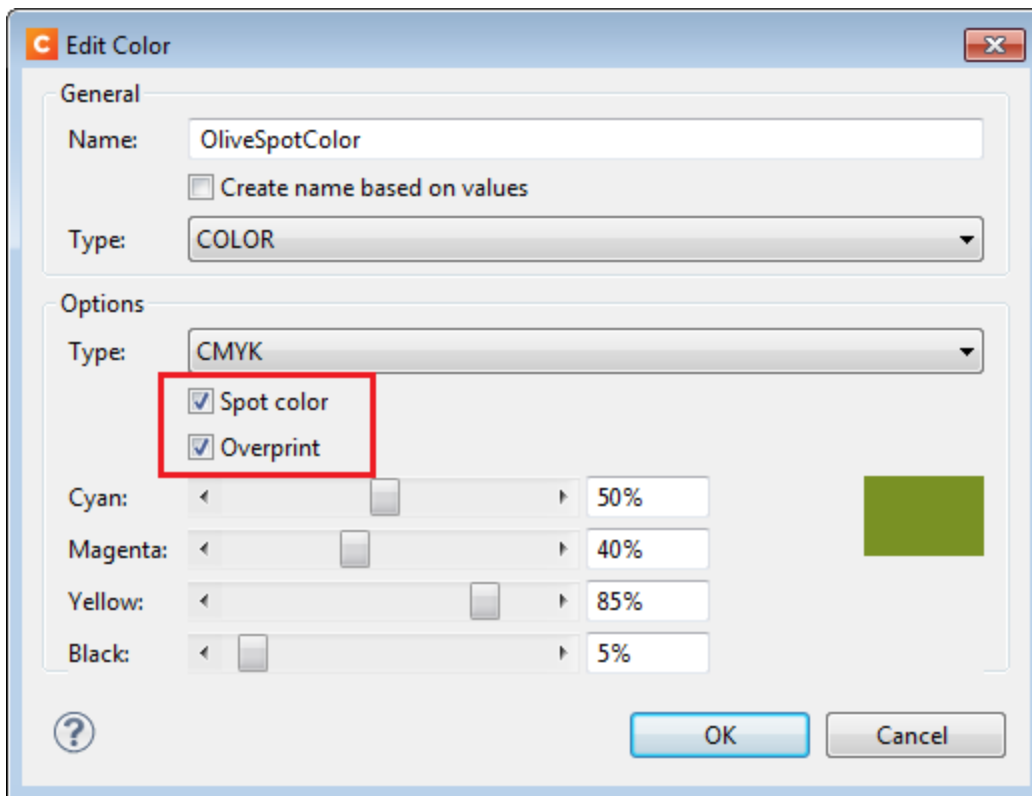
whether this will be an issue on your specific printer(s) or not. This issue will also be addressed in a later release.

## Overprint for Spot Colours

Overprinting certain content on top of other content is sometimes required. (SHARED-56743)  
For example:

- To deal with special print applications, such as applying special (invisible) inks that are intended to go on top of coloured areas, for instance printing UV ink or applying varnish to a certain area.
- To avoid mis-registration when printing black on top of coloured areas.

To support scenarios like these, Connect now supports Overprint for Spot Colours.



**Note:** Overprinting does not show on-screen in the Designer.

## Print Output

- **Mixplex support added**

An option has been introduced to omit empty back sides for the *Single* and *Last* sheet positions when Duplex is enabled, resulting in mixplex output. This helps in reducing costs in printing environments where page count or click-charging is applied. (SHARED-46965\55459)

- **Improved logging** of output generation. (SHARED-53367)



# Capture OnTheGo (COTG) Enhancements and Fixes

## New and improved COTG library

A new **jQuery** plugin variant of the COTG.js library introduces *events* and *options* for COTG widgets. These concepts greatly simplify event based programming. For example: it allows your code to set a date for a date field and retrieve the geolocation automatically on the drawing of a signature.

The COTG jQuery plugin is the successor of the *cotg-1.x.js* JavaScript library found in COTG forms based on the COTG Starter templates (v1.x). The main purpose of the COTG library is to link the COTG specific form controls with the hardware features of your mobile device. It also implements special controls like the **Fields Table** and makes field names unique for each row in a detail table. (SHARED-44058)

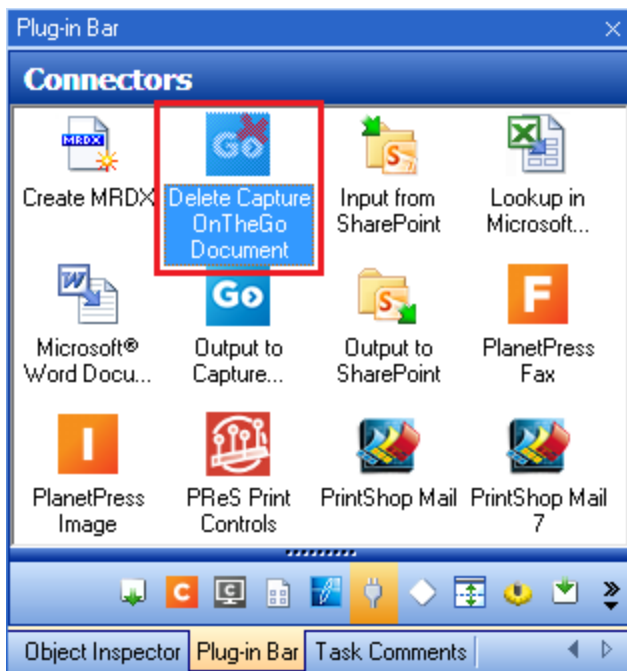
## Other COTG Improvements

- COTG Server selection added to **Send COTG Test** dialog. (SHARED-30002)
- **Unchecked checkboxes** now automatically submit a value of '0' (zero). (SHARED-50655)
- Add support for inserting form inputs and COTG inputs in the **Snippet editor** even when a `<form>` element is missing. (SHARED-55885)
- The Send COTG Test dialog now allows submission of **blank forms** for testing purposes. (SHARED-56001)
- The Camera Properties dialog has a new option to **enable/disable time stamping**. (SHARED-56991)
- The Geolocation Properties dialog now includes an option to **enable/disable the "High Accuracy" flag**. (SHARED-57004)

# Workflow 8.8 Enhancements and Fixes

## Deleting documents from the COTG repository

Many COTG customers create documents for which they don't want to set an expiry date (or a lifespan). They therefore set an expiry date far into the future to make sure the document remains in the repository. However, once the document has been filled they would like to remove it from the repository as soon as possible so it doesn't clutter the view of the repository. Accordingly, a new Workflow task now allows deletion of a document from the COTG repository, using just the ID. The ID is obtained when the document is first sent to the COTG server and could conceivably be stored in Workflow's Data Repository until you need it to delete the document. The task can be used either as a standard Action Task or as a Condition Task that is set to True when the deletion is successful. (SHARED-55313)

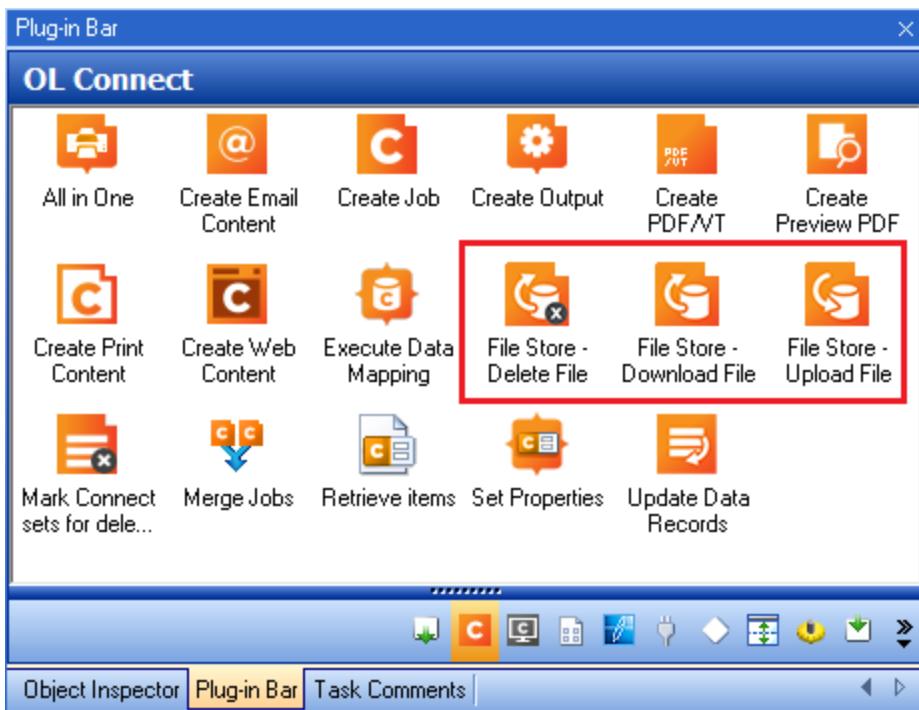


## Accessing the Connect Managed File Store

Sometimes you need a central location for storing and retrieving files. Some of those files might only be required for a brief period of time while others may have to be stored for longer periods. The usual approach is to create a folder somewhere on the Workflow system and use it as a file repository. But this solution is neither portable nor completely centralized since it relies on the characteristics of the specific system on which it is implemented.

The Connect Server has its own File Store which it uses for transient files. This File Store is managed by the Cleanup service who takes care of removing obsolete files when those files are not marked as permanent. This greatly reduces the amount of administration required to manage the files. We figured that since there is already a File Store and REST API calls to connect to it, why not turn that into an accessible feature for customer implementations?

With Connect 1.8, the Workflow 8.8 tool implements three new tasks that allow you to **Upload**, **Download** and **Delete** files in the Connect File Store.



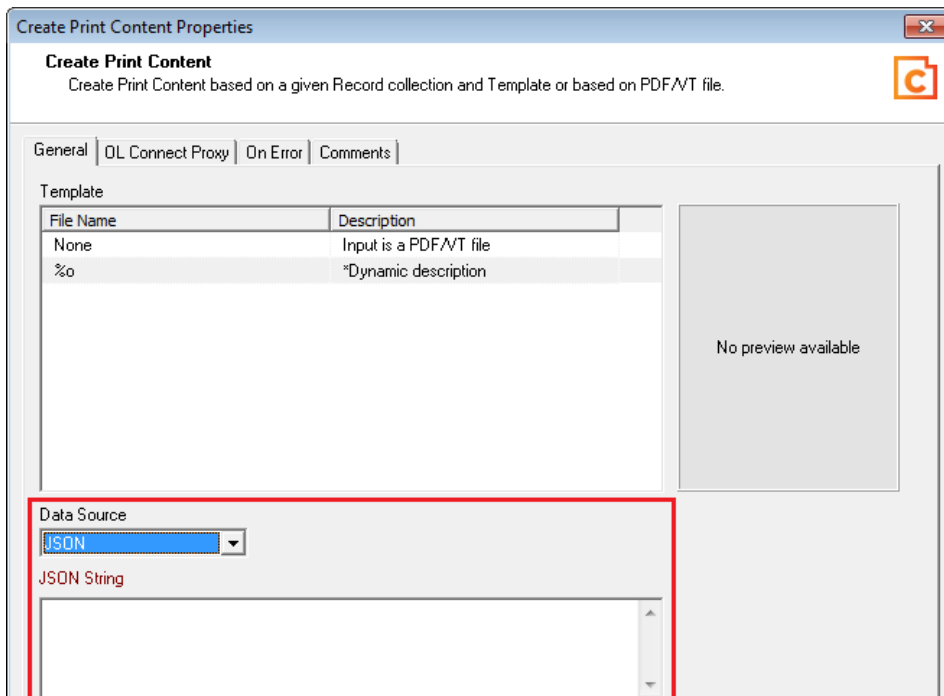
Files can be marked as permanent, in which case the Cleanup service won't touch them. The files can still be accessed through the REST API, which means web portals could potentially access the files directly without having to go through a Workflow process. The feature is therefore centralized and portable as it does not rely on any specific location or folder. (SHARED-57617)

## Generating all contents using JSON instead of data records

A number of customer workflows involve generating new versions of documents based on the original one created by Connect. For instance, a Delivery Note might come back with adjusted quantities on certain line items. Or a web-based status page might need to get updated with additional data. However in many cases, the original data must still be left untouched (for reprints, for instance).

The traditional approach to recreating the document would be to generate a new single data record with the new data and perform the usual data mapping/content creation/output creation operations. However, that means the data records are duplicated in the database and the entire operation takes slightly longer than it could.

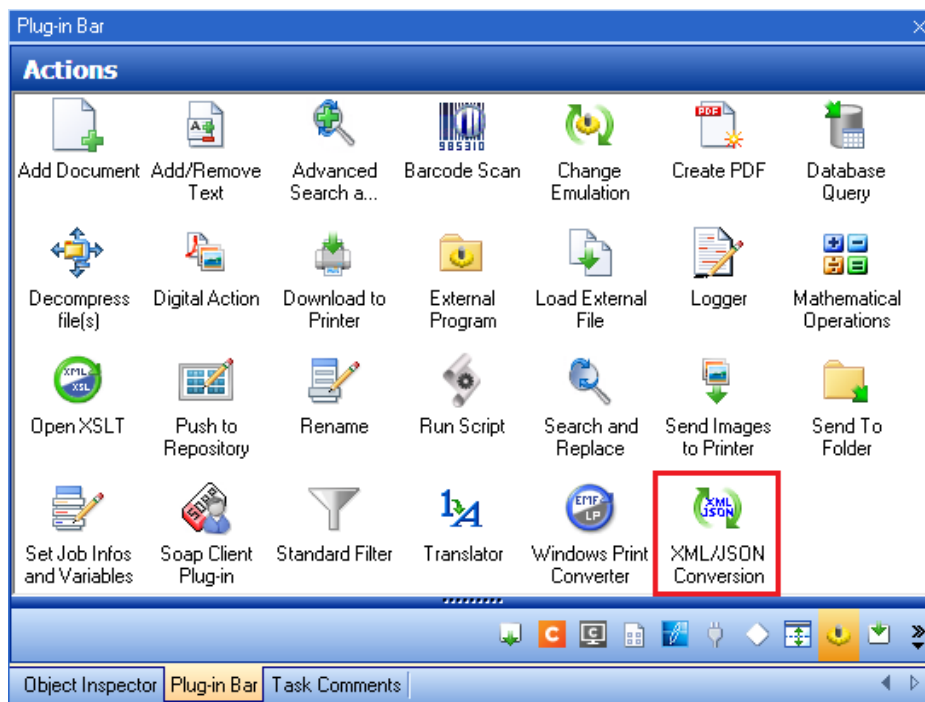
As of Connect Workflow 8.8 all *Connect Create Content* tasks have the option of using a JSON string as their data source instead of relying on the metadata generated from a data mapping operation.



This feature builds on the same kind of improvement that was implemented in Connect Workflow 8.7 with the *Create PDF Preview* task that provided lightning fast PDF Creation. The new feature makes it much easier to create Web or Email status pages that are delivered almost instantly. (SHARED-51086/55414)

## Converting XML to JSON and JSON to XML

XML is already one of the most popular data formats used with Connect. In many implementations, most notably web-based ones, XML is used to transfer or update information back and forth between Connect and Workflow. However, using XML means the DataMapper must be involved each time the data changes, which impacts performance, especially when all you want is to update a Status web page. With the new functionalities added to all *Content Creation* tasks allowing them to receive JSON as the data container, it is only natural that Workflow provide an easy way of converting data between JSON and XML formats.



The new task is simple to use: pick a destination format (XML or JSON) and the job file is immediately converted to that format. No scripting required! (SHARED-51089)

### Note

The JSON data can be used in other settings as well, not just for creating Connect content. For instance, converting a HTTP Request to JSON would allow a script to easily turn that JSON into an object and manipulate its properties with much more readable syntax than using `ExpandString()` on data selections.

## Full Timestamp entries added to Data Repository

Added full timestamps to the Repository to allow for more precise information on each key set. The *DateC* and *DateM* keys now both contain a full time stamp in the form of: YYYY-MM-DDThh:mm:ss.sZ.

Please consult the online Help for a full description of the Timestamp. (SHARED-52160)

## General Workflow fixes and enhancements

- Additional **Tooltips** added to interface. (SHARED-54981)
- Improved support for multiple IDs in **Retrieve items** task. (SHARED-53740)
- Repository access improvement and memory leaks fixed. (SHARED-56424)

# Known Issues

## Issues with Microsoft Edge browser

The Microsoft Edge browser fails to display web pages when the Workflow's CORS option (in the HTTP Server Input 2 section) is set to "\*". This issue will be resolved in a future release.

## Workflow - "Execute Data Mapping" - Issues with multiple PDFs

If a process has a *Folder Capture* step (but not the first input step) to capture multiple PDFs within a folder, followed by a *Execute Data Mapping* step to extract XML files for each PDF, only the first PDF file will be processed. The workaround is to put the Extract Datamapper in a Branch, just after the Folder Input. This issue will be fixed in a later release. (SHARED-59752)

## Installation Paths with Multi-Byte Characters

When installing the Chinese (Traditional or Simplified) or Japanese versions of Connect, if the user specifies an alternative installation path containing multi-byte/wide-char characters it can break some of the links to the Connect-related shortcuts in the Start Menu and cause an error to appear at the end of the installer. The workaround for the moment is to use the default installation path. The problem will be addressed in a later release.

## Switching Languages

Changing the language using the **Window>Preferences>Language Setting** menu option does not currently change all of the strings in the application to the selected language. This is a known issue and will be fixed in a later release.

In the meantime we offer the following workaround for anyone who needs to change the language:

1. Go to the .ini files for the Designer and Server Config:
  - C:\Program Files\Objectif Lune\OL Connect\Connect Designer\Designer.ini
  - C:\Program Files\Objectif Lune\OL Connect\Connect Server Configuration\ServerConfig.ini
2. Change the language parameter to the required one under Duser.language=en | es | de | fr | it | ja | ko | pt | tw | zh

Only one of the above language tags should be selected. Once saved, Connect will appear in the selected language at next start-up.

### **GoDaddy Certificates**

When installing Connect offline, dialogs allow installing the GoDaddy certificates. Most users should use the default settings and click **Next**. In some cases, however, this may not work correctly. For this reason those users should activate **Place all certificates in the following store** and then select the **Trusted Root Certification Authorities** as the target certificate store.

### **MySQL Compatibility**

After installing Connect 1.8 a downgrade to a Connect version earlier than Connect 1.3 or to a MySQL version earlier than 5.6.25 is not seamlessly possible. This is because the database model used in Connect 1.3 and later (MySQL 5.6) is different to that used in earlier versions. If you need to switch to an older version of Connect / MySQL, it is first necessary to remove the Connect MySQL Database folder from "%ProgramData%\Connect\MySQL\data" before installing the older version.

### **PostScript Print Presets**


The print presets for PostScript were changed from Version 1.1 onwards meaning that some presets created in Version 1.0 or 1.0.1 may no longer work.

Any PostScript print preset from Version 1.0 that contains the following will not work in Version 1.8: \*.all[0].\*

Any preset containing this code will need to be recreated in Version 1.8.

### **Available Printer Models**

Note that only the single Printer Model (Generic PDF) will appear on the **Advanced** page of the **Print Wizard** by default.

To add additional printer models click on the settings  button next to the Model selection entry box.

Note that the descriptions of some of the printers were updated in version 1.2 meaning that if you had version 1.n installed, you may find that the same printer style appears twice in the list, but with slightly different descriptions.



For example the following printer types are actually identical:

- Generic PS LEVEL2 (DSC compliant)
- Generic PS LEVEL2 (DSC)

## External Resources in Connect

There are certain limitations on how external resources can be used in Connect. For example if you want to link a file (e.g., CSS, image, JavaScript etc.) from a location on the network but you do not want to have a copy of the file saved with the template you need to do the following:

1. The resource must be located where it can be accessed by all Servers/Slaves run as users. Failure to do this will cause the image to appear as a Red X in the output for all documents which were merged by engines which could not access the file. The job will terminate normally and the error will be logged.
2. The file must be referenced via a UNC path e.g.,  
file:///w2k8r2envan/z%20images/Picture/Supported/JPG/AB004763.jpg
  - UNC paths are required because the services will be unable to access mapped network drives (Windows security feature).
  - The engine processing the job will look on the local file system for the direct file path leading to the “resource not found” issue mentioned above.

### Warning

Important Note: The Designer itself and Proof Print do not use processes that run as services and they may find local files with non-UNC paths which can lead to the false impression that the resources are correct.

## Using Capture After Installing Workflow 8

If PlanetPress Connect Workflow 8 is installed alongside PlanetPress Suite Workflow 7, Capture can no longer be used within Workflow 7. The plugins are now registered uniquely to Workflow 8 and the Messenger for Workflow 7 is taken offline. It is only possible to use Capture from PlanetPress Connect Workflow 8 thereafter.

## Capturing Spool Files After Installing Workflow 8

If PlanetPress Connect Workflow 8 is installed alongside PlanetPress Suite Workflow 7, the PlanetPress Suite 7 option to capture spool files from printer queues will no longer function. The solution is to use PlanetPress Connect Workflow 8 to capture spool files from printer queues.

## Colour Model in Stylesheets

The colour model of colours defined in a stylesheet can sometimes change after editing the stylesheet. This is a known issue and will be addressed in a subsequent release.

## Image Preview in Designer

If in the Windows Internet settings (**Connection Settings > LAN configuration**) a proxy is enabled, but "Bypass proxy settings for local addresses" is not checked, the image preview service, conversion service and live preview tab in the Designer will not work and exhibit the following issues:

- Images will be shown as 0 size boxes (no red 'X' is displayed)
- Live preview does not progress, and when re-activated reports "browsers is busy"

To fix the issue you must check the "Bypass proxy settings for local addresses" option.

## MergeWeaver Engines when Printing

The print operation in the Designer will automatically detect whether the Merge\Weaver engines are available and display a message for the user to retry or cancel if not. Once the MergeWeaver engine becomes available and the user presses retry the print operation will proceed as normal. This message can also occur in the following circumstances:

- If the server is offline and you are not using Proof Print
- On some occasions before the Print Wizard opens

## REST Calls for Remote Services

The Server will now accept REST calls for all remote services and will make commands wait indefinitely until the required engines become available. The Server will log when it is waiting for an engine and when it becomes available. Note that there is no way to cancel any commands other than stopping the Server.

## **Print Content and Email Content in PlanetPress Workflow**

In PlanetPress Workflow's Print Content and Email Content tasks, the option to Update Records from Metadata will only work for fields whose data type is set to String in the data model. Fields of other types will not be updated in the database and no error will be raised. This will be fixed in a later release.

## **Print Limitations when the Output Server is located on a different machine**

The following limitation may occur when using the Print options from a Designer located on a different machine to the Output Server:

- The file path for the prompt and directory output modes is evaluated on both the client AND server side. When printing to a network share it must be available to BOTH the Designer and Server for the job to terminate successfully.
- The Windows printer must be installed on both the Server and Designer machines.
- When printing via the Server from a remote Designer, the output file remains on the Server machine. This is remedied by selecting "Output Local" in the Output Creation configuration.

## **VIPP Output**

Some templates set up with landscape orientation are being produced as portrait in VIPP. It can also sometimes be the case that text and images can be slightly displaced. These are known issues and will be addressed in a later release of Connect.

# **Previous Releases**

## **Overview**

This document provides an overview of the new features and enhancements in PlanetPress Connect 1.7.1 and PlanetPress Workflow 8.7.

### [Installing PlanetPress Connect 1.7.1 and PlanetPress Workflow 8.7](#)

- PlanetPress Connect is released as a 64 Bit version only (with the exception of the Workflow, Fax, Search and Imaging modules).

- Full details on installing and licensing PlanetPress Connect and PlanetPress Workflow can be found in the online help in the installer.
- Note that both PlanetPress and PlanetPress Connect Workflow come with 30 day trial licenses by default.

### Upgrading from PlanetPress Connect 1.1

In order to upgrade from Connect Version 1.1 to Version 1.7.1 via the Update Manager, it is necessary to install a new version of the Objectif Lune Update Client. The next time you run your current Update Client it will show that there is an update available for itself. Simply click on the download button in the dialog to install the new version of the Update Client. Note that it is no problem to run the update while the Client is open. It will automatically update itself.

Once you have done this, PlanetPress Connect 1.7.1 will become available for download.

From Connect Version 1.2 onwards, the newer version of the Update Client was included with the Connect installation.

### Updating stand-alone Workflow Messenger installations

If Workflow Messenger were installed stand alone, with no other Workflow components installed, the Update Client will be unable to find the Messenger component and thus it will not automatically update to the Workflow 8.7 version. To get around this, download and run the Workflow 8.7 installer manually.

### Print Only Version

A Print Only license is available with version 1.7.1 of PlanetPress Connect which allows legacy PlanetPress Suite 7 customers on OL Care to upgrade to Connect for a minimal fee. The license allows regular printing via the Print Wizard but runs Email and Web output in demo mode. For more information, please contact your local OL Customer Care or Sales team.

### Templates Used in Workflow

For improved performance we recommend re-saving Workflow templates set up in the previous versions to run with PlanetPress Connect 1.7.1\Workflow 8.7.

## Reduced Memory Version

### **Note**

This is **not** recommended for production.

It is now possible to install PlanetPress Connect on a machine with a minimum of 2 GB of RAM. The PlanetPress Connect Designer will automatically detect whether it has been installed on a machine with less than 4 GB of RAM and default to only using one internal Weaver and one internal merge engine on that system. The Server will also run using internal engines.

# Connect 1.7.1 General Enhancements and Fixes

## Template Reports added to Connect

Generate a report in PDF format containing the most important information about your template. The report lists contexts, sections, master pages, scripts, the data model, graphic files, and any other resources used, along with their properties. This report can be added as part of your project documentation.


The report is created using Connect technology, and it generates an XML file and thumbnails, allowing you to create your own custom report structure and corporate styling. This can be achieved by altering the underlying Connect Template and Data Mapper configuration.

The following screen-shot shows an extract from a sample report that was created for the *OL Mobile Letter* template:

### Print Context

#### Section 1

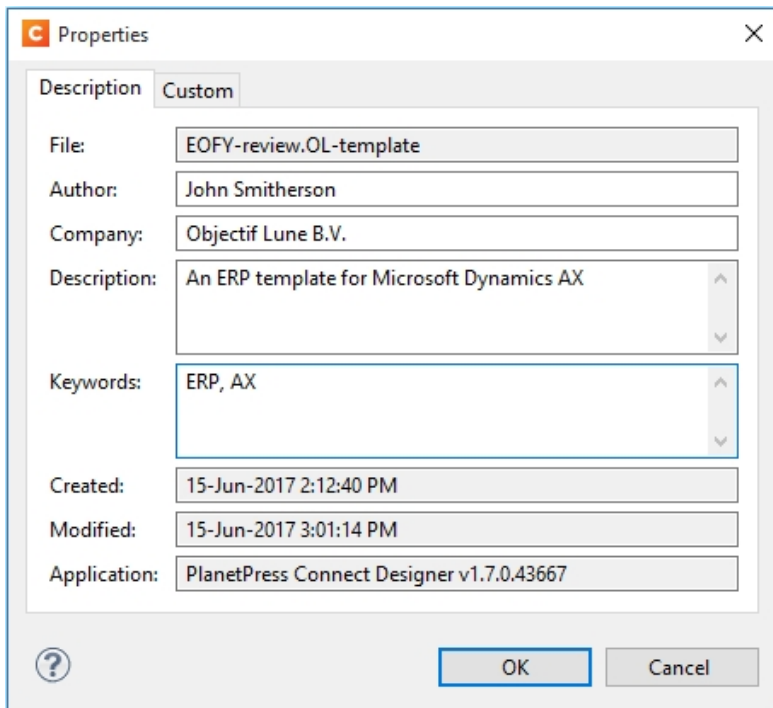
| Properties  |                                                         |
|-------------|---------------------------------------------------------|
| Name        | Section 1                                               |
| Page size   | Letter                                                  |
| Dimensions  | 8.5in x 11in                                            |
| Orientation | portrait                                                |
| Margins     | Top: 1.5in, Right: 0.75in, Bottom: 0.75in, Left: 0.75in |
| Bleed       | Top: 0in, Right: 0in, Bottom: 0in, Left: 0in            |



| Includes                 | Type | Location                     |
|--------------------------|------|------------------------------|
| context_all_styles.css   | css  | css/context_all_styles.css   |
| context_print_styles.css | css  | css/context_print_styles.css |
| default.css              | css  | css/default.css              |

## Document Properties

Document Properties can now be added to both Templates and Data Mapper Configurations. This allows you to specify properties such as the document author, the customer name and other important references. You can also add custom key/value pairs. The respective properties can be retrieved in scripting and are thus available as content in your documents. The information is also included in the *Template Report* feature. (SHARED-47780)



The screenshot shows a 'Properties' dialog box with the following fields and values:

| Field        | Value                                     |
|--------------|-------------------------------------------|
| File:        | EOFY-review.OL-template                   |
| Author:      | John Smitherson                           |
| Company:     | Objectif Lune B.V.                        |
| Description: | An ERP template for Microsoft Dynamics AX |
| Keywords:    | ERP, AX                                   |
| Created:     | 15-Jun-2017 2:12:40 PM                    |
| Modified:    | 15-Jun-2017 3:01:14 PM                    |
| Application: | PlanetPress Connect Designer v1.7.0.43667 |

## Stability improvements

- Improvements made to the Clean-up service. In some production environments the database Clean-up could not keep pace with database growth, leading to the database gradually filling up. This has been fixed through an improved internal database structure and more efficient queries and deletions. (SHARED-46465/52345)
- The PlanetPress Connect server was experiencing communication problems with the engines in some circumstances, after data mapping or content creation errors were encountered. These issues have now been resolved. (SHARED-55165)

## "Enhance with Connect" option added for PDF files in Windows Explorer

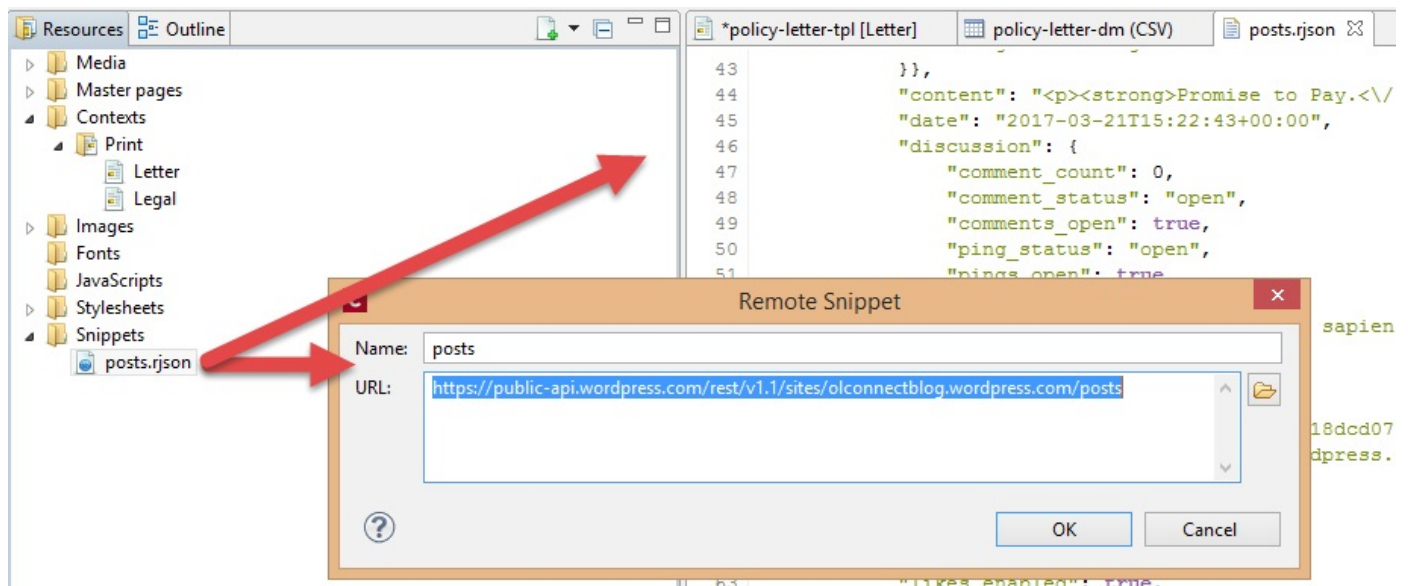
A Windows Explorer context menu entry "**Enhance with Connect**" has been added for PDF files. When a user selects this context menu entry, PlanetPress Connect Designer opens with a prefabricated template, that uses the selected PDF file as the background. (SHARED-15350/47156)

## Support added for Remote HTML and JSON Snippets

In PlanetPress Connect 1.7.1 we introduce the concept of remote snippet resources. These snippet entries have a Name and URL property (e.g., the hyperlink to the endpoint) and reside in the Snippets folder located in the Resources panel.

In scripts these snippet entries are referenced just like regular snippets, e.g., `loadhtml ('snippets/my_content.rhtml')` or `loadjson ('snippets/posts.rjson')`. Note the "r" in the file extension.

Having the snippet entry in the **Snippets** folder within the **Resources** folder allows us the simplest overview of the resources used. In previous versions this behaviour would have had to be captured in script and therefore would not have been directly visible as part of the resources. This new approach greatly simplifies maintenance of the URL, as it can now simply be updated in the Resources panel rather than by browsing through all the scripts. (SHARED-42314/52591)

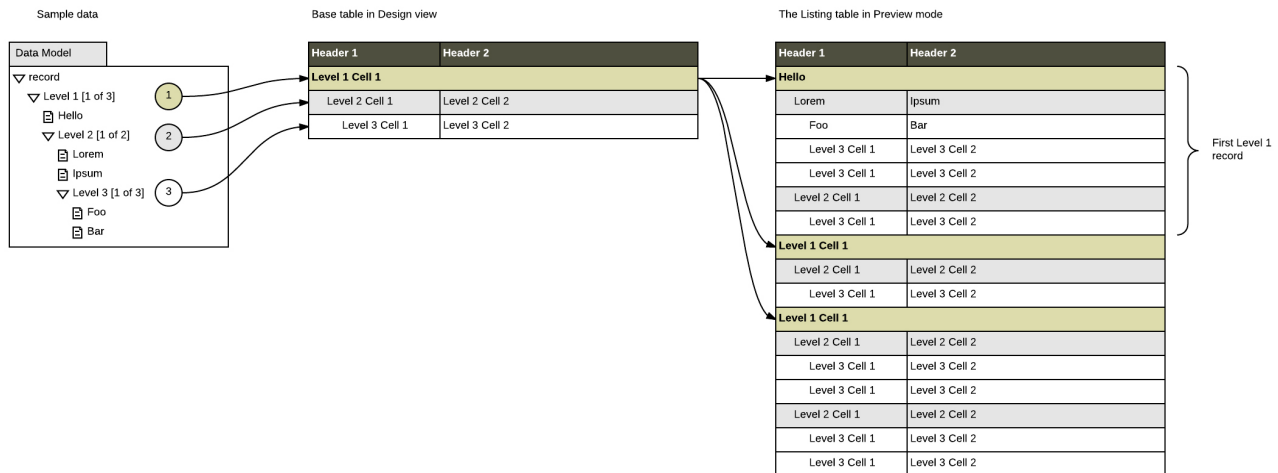




## Handling Nested Detail Data

Simplifying the handling nested detail data has been on our agenda for some time. As part of our research into this we have looked at an approach that repeats table rows for nested detail data. This doesn't create HTML tables in HTML tables but rather clones a base row specified for each level.

Consider the following image:



At this stage there is no user interface to configure this type of dynamic table but in a separate Technical article. It can be achieved by setting some HTML attributes in the *Source* view and adding scripting to populate the cells.

A user interface (table wizard) to set things more elegantly will be introduced in a future version, along with additional functionality, such as subtotal calculations.

As we are rather excited about this approach we wanted to share the current state with you in PlanetPress Connect 1.7.1. (SHARED-43047)

## Installer improvements

- The PlanetPress Connect 1.7.0 installation did not work on machines running Windows 10 build 1703 (i.e. the "Creators Update", released March 2017). This has been fixed for PlanetPress Connect 1.7.1. (SHARED-56800)
- The **silent installation** process has been enhanced, and now supports the following:
  - **Setting the repository.** This can be configured via the "*product.repository*" entry in "*install.properties*". (SHARED-17841)
  - **Selecting a dedicated locale** (language and country code) for the Connect applications. These can be configured by the "*user.language*" and "*user.country*" entries. (SHARED-18381)
  - Improved reporting of Silent Installation success or failure. (SHARED-17723)
- Microsoft SQL server connection settings added to Connect Installer. (SHARED-36866)
- The Update Client will now run the installation in the same language as the original installation. (SHARED-37868)
- Installer has been made more robust, and will now continue (with warning messages, if applicable) when it encounters any of the following scenarios:
  - If Server start-up was unsuccessful during installation. (SHARED-39398/46837)
  - If no Database connection could be established. (SHARED-39400)
  - The Installer now checks if the OL Connect MySQL service is in the proper state and resident in the expected folder. If is not, instructional warning messages are now displayed. (SHARED-40309/45431)
  - If Connect folders that should have been deleted were found upon re-installation. (SHARED-41420)

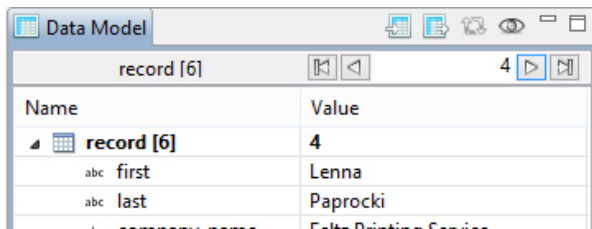
# Connect 1.7.1 Designer Enhancements and Fixes

## Edit and Save CSS, HTML, JavaScript and JSON files within the Designer

Ever needed to quickly edit an external CSS, HTML, JavaScript or JSON file? The PlanetPress Connect 1.7.1 Designer now allows you to open and save these file types via the **File** menu. (SHARED-42094)

## Data Model Panel Enhancements

Various enhancements have been made to the **Data Model** panel. The browse options of the main record are now *sticky* and do not move out of view when working with a large number of data fields. An eye icon has been added to the toolbar, and is used to toggle the visibility of the *ExtraData* field. In addition, you can select and group multiple fields in order to collapse them out of view (and expand them back, obviously), which is particularly useful when dealing with large data models that force you to constantly scroll up and down to bring a specific field into view. (SHARED-45370/54106)



The Data Model panel has also been enhanced to allow alphabetical sorting of detail tables. (SHARED-47169)

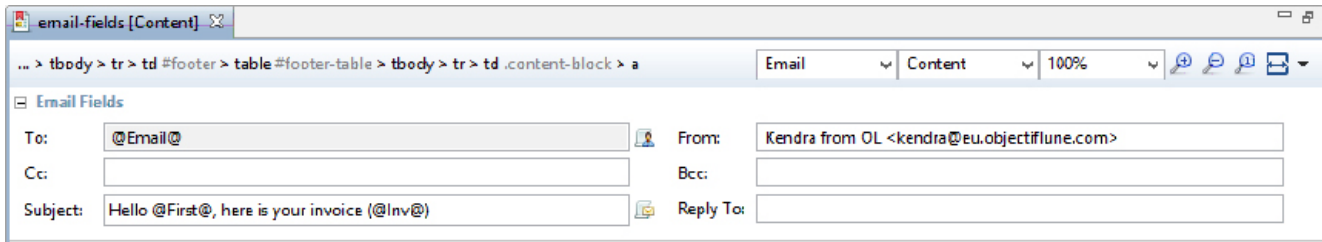
## Simpler Invocation of Email Script Wizards

Invoke email related script wizards simply by clicking the labels in the email information bar. (SHARED-47329)

## Simplified Email fields User Interface

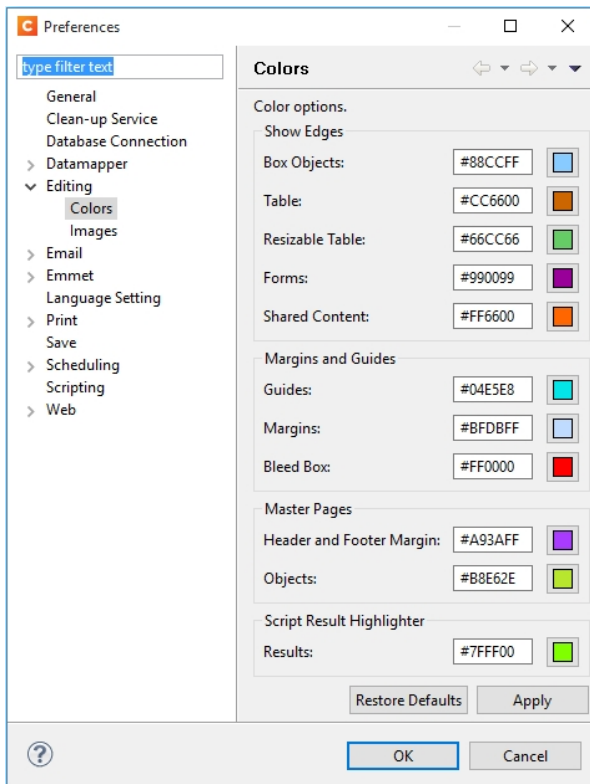
Create *To*, *CC*, *BCC*, *From* and *Reply To* email scripts by dragging and dropping a data field to the respective input field or type a static address directly in the input fields. (SHARED-9178)

Type the subject in the Subject email field and drag and drop data field(s) to positions in that string to make a personalized email subject without any scripting. (SHARED-51475)



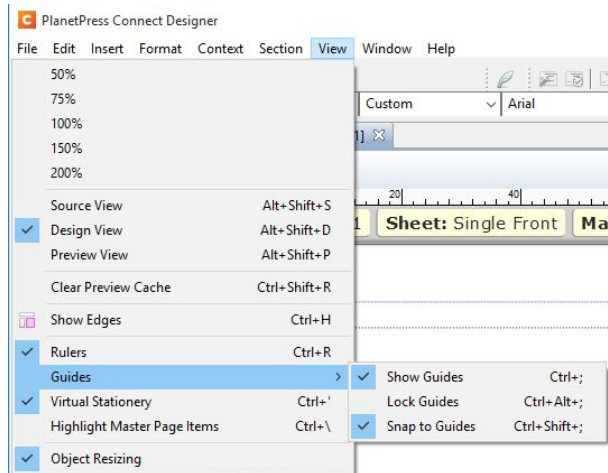
## Improved Customization of the Designer interface

Customize your interface by selecting your own colours for object edges, margins, guidelines, etc. (SHARED-49841)



## Guideline behaviour improved

Along with visible/invisible settings, Guidelines can now be locked in place or set to snap to objects, using the new **Guides** option in the View menu. (SHARED-47159).



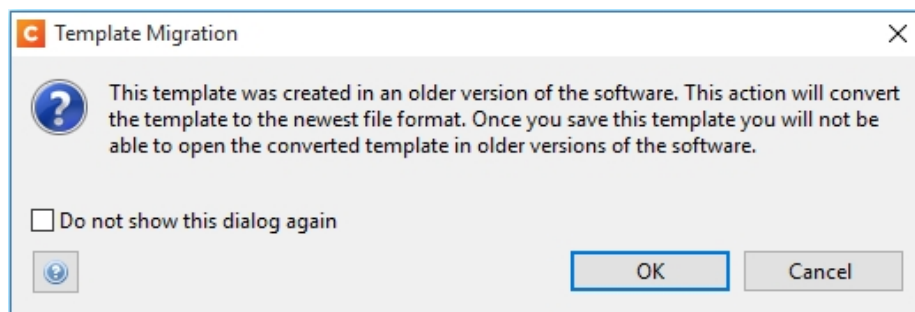
## Warning now displayed when opening templates created in an older version

When PlanetPress Connect opens an older template file it is automatically migrated to the template structure of the current version. Saving the file in the new version would thus update the file format and prevent the document opening in an older version.

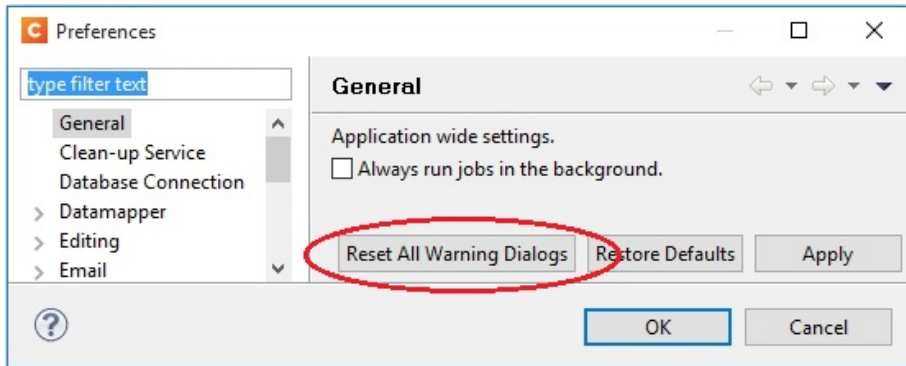
A warning is now shown when opening a Template created in an older PlanetPress Connect version, allowing you the chance to save the Template to a new file, leaving the original intact. (SHARED-51912)

## Turn Warning dialogs Off/On

A "**Do not show this warning again**" check-box option has been added to many PlanetPress Connect Warning dialog boxes:

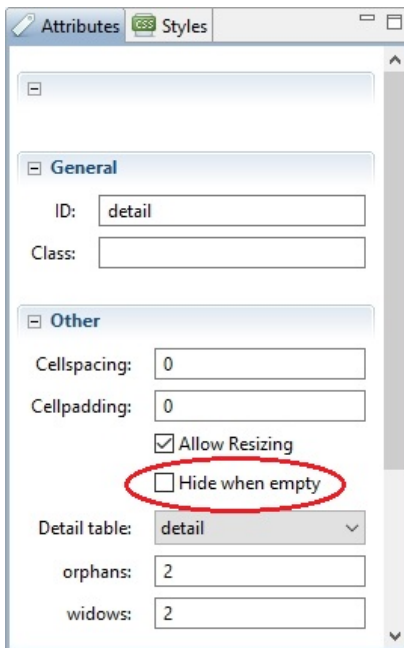


These Warnings can be switched on again at any time thereafter, via the "**Reset All Warning Dialogs**" button in the General Preferences dialog. (SHARED-16962)



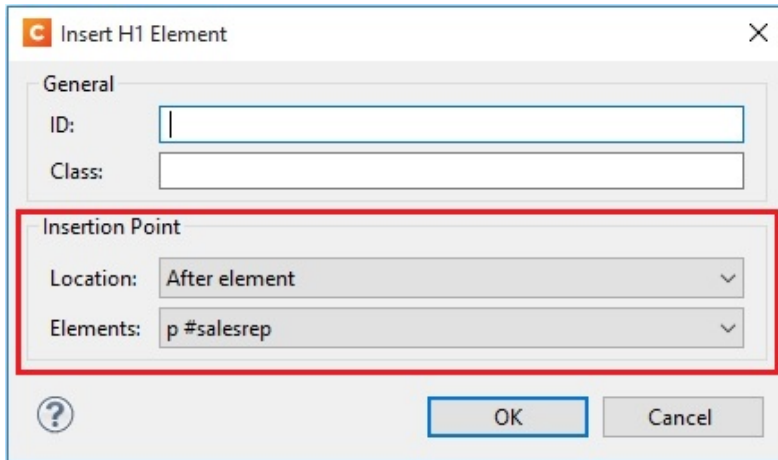
Option to automatically Delete a dynamic table when the table is empty

An option has been added to allow you to automatically delete a dynamic table when the data table is empty. To do so, select the entire table, and then tick the "**Hide when empty**" checkbox in the Attributes panel. (SHARED-43537)



## Replace elements with data-insert-location when inserting HTML elements

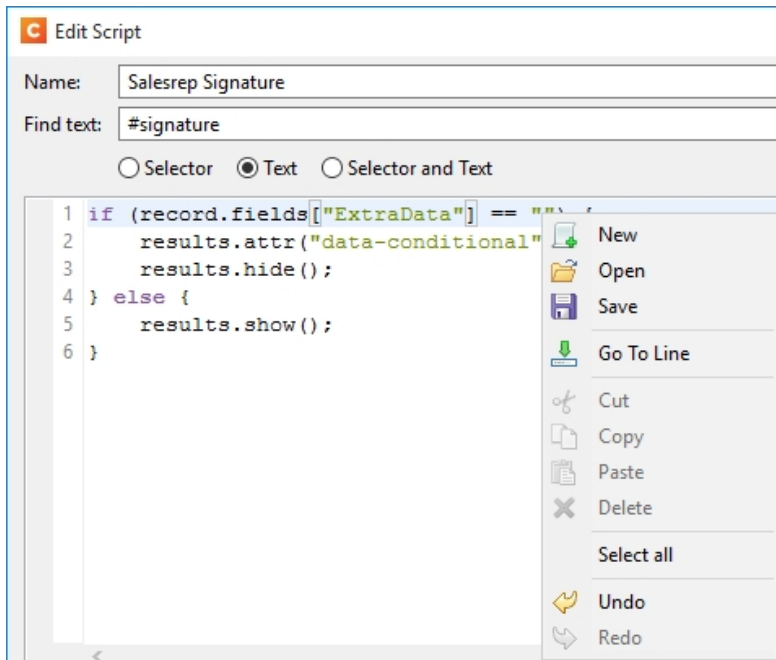
When inserting an element from an **Insert dialog**, Connect now checks the *data-insert-placeholder* attribute. The value of the attribute is then used to set the default value for the *Insert Location* option within the Insert dialog. If the attribute is not found, things behave as in previous versions.



This ticket also introduces the Replace option for the Insert Location drop down. When selected the to-be-inserted element(s) will replace the currently selected element(s). (SHARED-52369)

## Scripting improvements

- Context menu added to the **Edit Script** dialog. (SHARED-45381)



- **Find and Replace** functionality has been added to Script editors. (SHARED-48424)
- New menu option to **rename** Scripts or Folders has been added to the Context Menu within the Script panel. (SHARED-48607)
- Support added for **copy and paste** of folders and scripts within the Scripts panel. (SHARED-49299)
- The JavaScript **parseInt()** method now defaults to using base 10 arithmetic rather than base 8, as defined in the ECMAScript specifications. (SHARED-49010)



## General Designer improvements

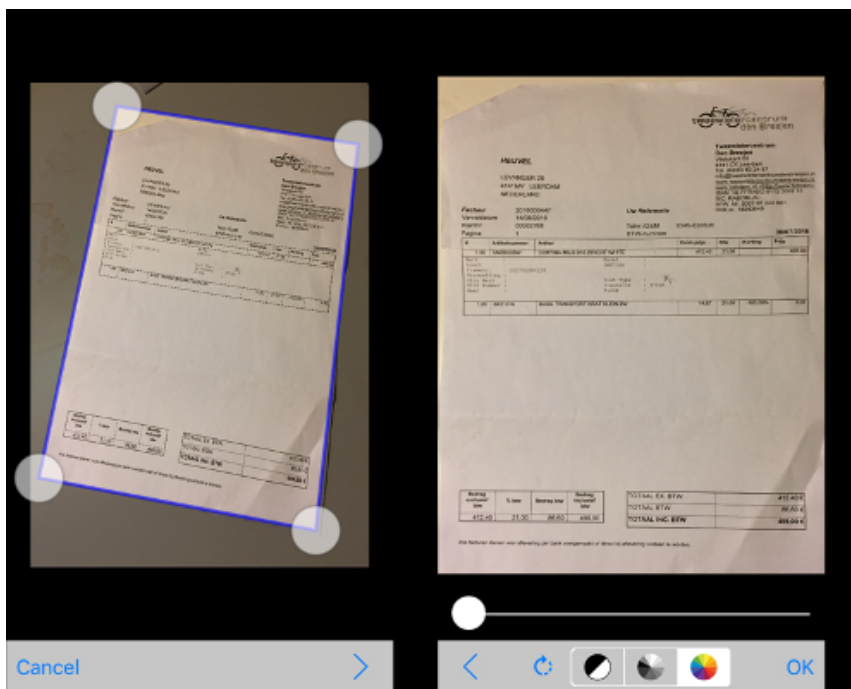
- **Duplicate and Delete line(s)** using shortcuts in the Stylesheet, JavaScript and HTML editors. Use `Ctrl+D` to **duplicate** and `Ctrl+Shift+D` to **delete** the currently selected lines. (SHARED-46928)
- Entering **geometry values** without stating a specific unit type will now automatically assign the default unit type to the entry. (SHARED-50656)
- When **deleting an element** (such as a Barcode or a Chart) on a page, a check will now be made for associated scripts. If any are found, the deletion step will provide an option to delete those scripts as well. (SHARED-45675)
- An "**All Files (\*.\*)**" filter was added to Save/Save-as dialogs. (SHARED-28237)
- **Icons** have been added for JS and CSS files in the Includes dialog to make it easier to distinguish between local and remote resources. (SHARED-47936)
- **Abs box** grippers and borders now display at a consistent thickness regardless of zoom level. (SHARED-50175)
- Support added for **dynamically setting the media background image** and its options via a Control Script. This only works for PDF files residing in the template at the moment. (SHARED-53524)

## Web form improvements

- Includes (JavaScript and CSS) can now be set for the entire Web context, rather than just per single web pages. These files are automatically linked to all web page sections and act as **global includes**. This is ideal when working with framework and library files that are used by all web page sections (for example jQuery or Foundation). One can still add additional includes on a per web page basis, if desired. (SHARED-48708)

## Capture OnTheGo (COTG) improvements

- **Two new form inputs** have been introduced to facilitate the retrieval of the document ID and the store ID. (SHARED-53987/54054)
- Improvements made to **updating the COTG library** within existing templates. The user will now be prompted as to whether they wish to switch to the new version or not. (SHARED-46920)
- Ability added to **insert dummy data** for the form inputs (including special COTG inputs such as the signature) upon retrieving the Job Data file from within the Designer. (SHARED-48676)
- Workflow "**Output to COTG**" task can now be run as an action and return Document IDs. (SHARED-48951)
- **Deskew (straighten) pictures.**  
A new option is added to the Camera Properties allowing the user of the COTG app to deskew or straighten images taken with their mobile device. Deskewing optimizes the image for further processing of the image, such as through an OCR process. (SHARED-53982)



# Connect 1.7.1 DataMapping Enhancements and Fixes

## DataMapper can now fetch or update data from remote sources

New in PlanetPress Connect 1.7.1 is the ability to create an XMLHttpRequest object (aka XHR) in DataMapper scripts in order to issue REST/AJAX calls to external servers. This feature allows the datamapping process to complement the extraction process with external data, including data that could be provided by a HTTP process in Workflow.

For instance, the DataMapper could issue calls to a Workflow process that retrieves certain values from the Workflow's Data Repository.

Also, one could imagine a DataMapper postprocessor that writes the results of the extraction process to a file and then uploads that file to a Workflow process. (SHARED-43502)

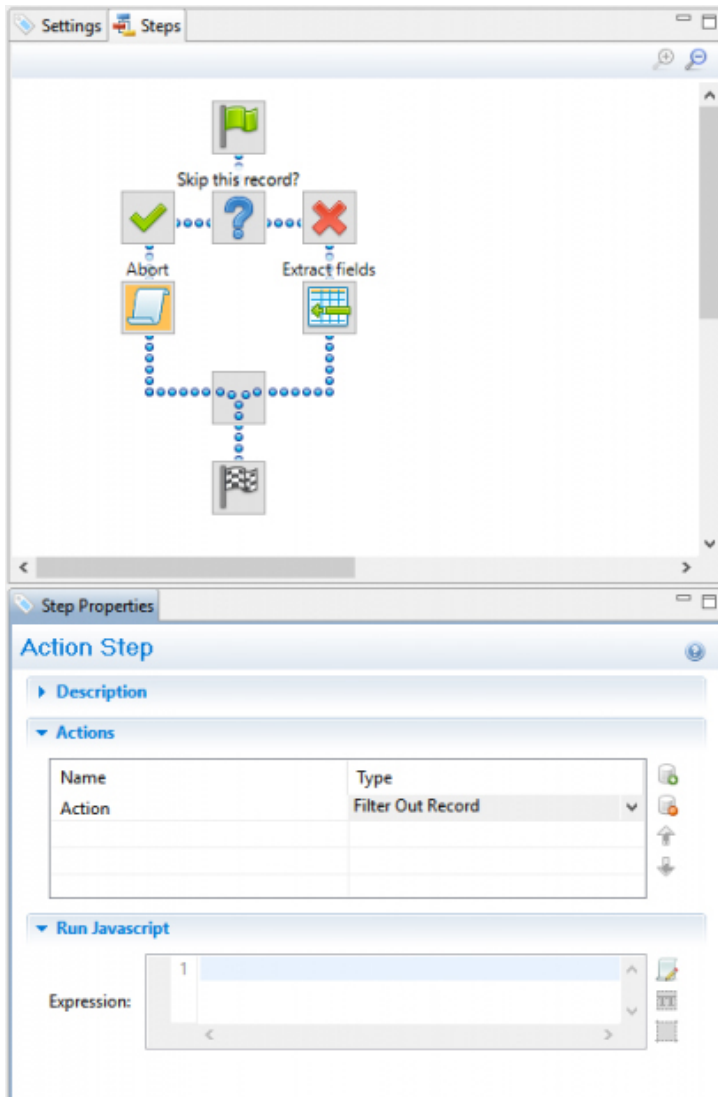
### Tip

As always with powerful features like this one, you need to be careful how you design your solution. For example it is possible to create an endless circular process, whereby a Workflow process calls a data mapping process, which in turn calls the same Workflow process, which calls the DataMapper again, and so on.

## Skip over Source Data Records

Another very important new feature implemented in the DataMapper is the ability to skip over some of the source data records without writing anything to the database. This allows you to quickly filter out some records from the data source without having to extract them to the database first, while still extracting others. (SHARED-24548)

Let's say for instance that your data source contains postal addresses from many countries, but you only want to extract the data for Portugal. You can now create a condition that examines the country field to determine if the source record is for a country *other* than Portugal. If the condition is *False* (i.e. the country IS indeed Portugal), you can extract the data as per usual. But in the *True* branch of the condition (i.e. the country *is anything but* Portugal), you can now add an Action Step and specify the new **"Stop processing record"** action type. This basically discards this data record and instructs the DataMapper to immediately skip to the next source record.



This yields two immediate and major benefits:

- Data Extraction is much faster since you are only extracting the records you actually want
- The database will not be cluttered with useless records (potentially numbering in the thousands) that you were not going to use anyway. As a consequence, the automated clean-up process will have much less work to perform when the time comes to delete obsolete entries from the database. This should result in a lighter workload and better overall performance.

### Note

If you stop processing any record *after* you've already extracted some data from it, then

the record will still be stored in the database, with un-extracted fields being assigned whatever default value (if any) you defined for them. So if your goal is to completely prevent unwanted records from being stored in the database, you should make sure to implement your filtering conditions early in the data mapping process.

#### Improved naming of default fields

The default field naming scheme has been enhanced to allow duplicate field names to be generated automatically as long as they are on different levels (or in different detail tables). This is especially useful with XML data sources where field names (e.g., ID, NAME) are often re-used through different levels of the structure.

With the new, more flexible naming scheme, the DataMapper checks for duplicates at the same level before deciding whether or not to create or increment a numeric suffix that is appended to the field name. (SHARED-42645)

#### Improvements made to XML File Processing

Issues were encountered with repeated nodes in XML datasets. In XML when adding an Extraction step, the XPath was not generated with an index, which resulted in only the first node being returned. This has been fixed, and repeated nodes are now catered for. (SHARED-28107/32238)

#### Improved support for Multi-Byte Encoding

Added support for byte based positioning in addition to the existing character based positioning for MultiByte (Big5, GB18030, UTF-8 and Shift-JIS) text files. (SHARED-53174)

## General DataMapper Enhancements and Fixes

- **XML Wizard**: option added to extract Attributes and to set boundaries on Attribute changes. (SHARED-42251)
- Improved support for **UNC paths** to image files. (SHARED-44316)
- The **Extradata** fields are now available in the DataMapper to more easily allow setting of field default values. The display of the Extradata fields can be toggled on or off directly from the Data Model panel. (SHARED-51426)
- New **data.findRegExp()** function added. This function is similar to the existing data.find() function, but with Regular Expression support added. (SHARED-51694)
- Repeat Step conditions can now be set to **evaluate operands as integers or as strings** to make it easier to compare numeric values without having to cast them first. (SHARED-49786)

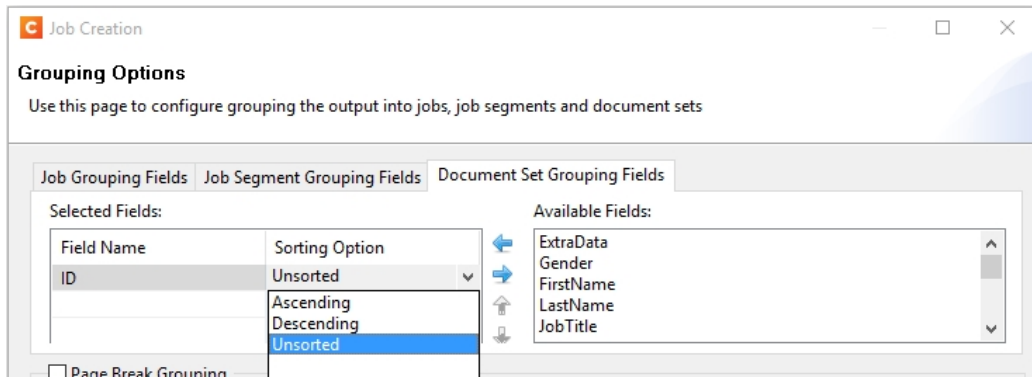
## Connect 1.7.1 Output Enhancements and Fixes

### Grouping With and Without Sorting

Sometimes the data used for generating documents is already pre-sorted, but you may still need to group documents into sets or segments. In those cases, the grouping process should not reorder the documents. This has now been implemented in PlanetPress Connect 1.7.1.

Consider the following example: data has been pre-sorted for postal sorting, which means that documents for the same customer will also be in consecutive order in the job (assuming a customer has a single postal address). If we want documents for the same customer to go into the same job, we can use grouping to create document sets and we might use the customer number for this grouping. When the customer number changes, we want a new document set to begin. If grouping by customer number also sorts by customer number, our pre-sorted order will get messed up.

The Job Creation settings have been improved to allow this kind of grouping. Sorting ascending, descending or not sorting at all can be set per field used for grouping. (SHARED-45125)



This means that, apart from straightforward cases where we are grouping with or without sorting, it is also possible to create combinations where some fields do alter the sort order and others have no effect.

Please note that grouping without sorting also means that any documents that have the same value for the same grouping field (i.e., customer number in the example above), but which are not consecutive in the input data, will not end up in the same group.

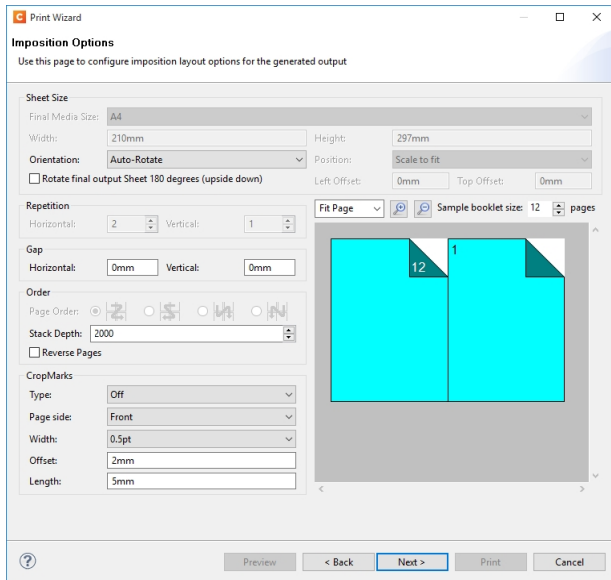
The settings for **Grouping** are available both in the **Job Creation Settings** dialog and the **Advanced** mode of the **Print Wizard**.

#### Progress of External Sort now displayed

When using an external sort, there was no feedback about how the external sort program was progressing. A new dialog control has now been added to Connect which displays the progress of the external sort in real time. (SHARED-53601)

#### Improvements made to Imposition Options dialog

The settings page for Cut and Stack Impositioning has been improved to show a sample of the chosen imposition settings.



Additionally, some settings on the Imposition Options page affect the way that booklets are created. These settings are now editable, so settings such as the gap between pages can now also be set for booklets. (SHARED-31097)



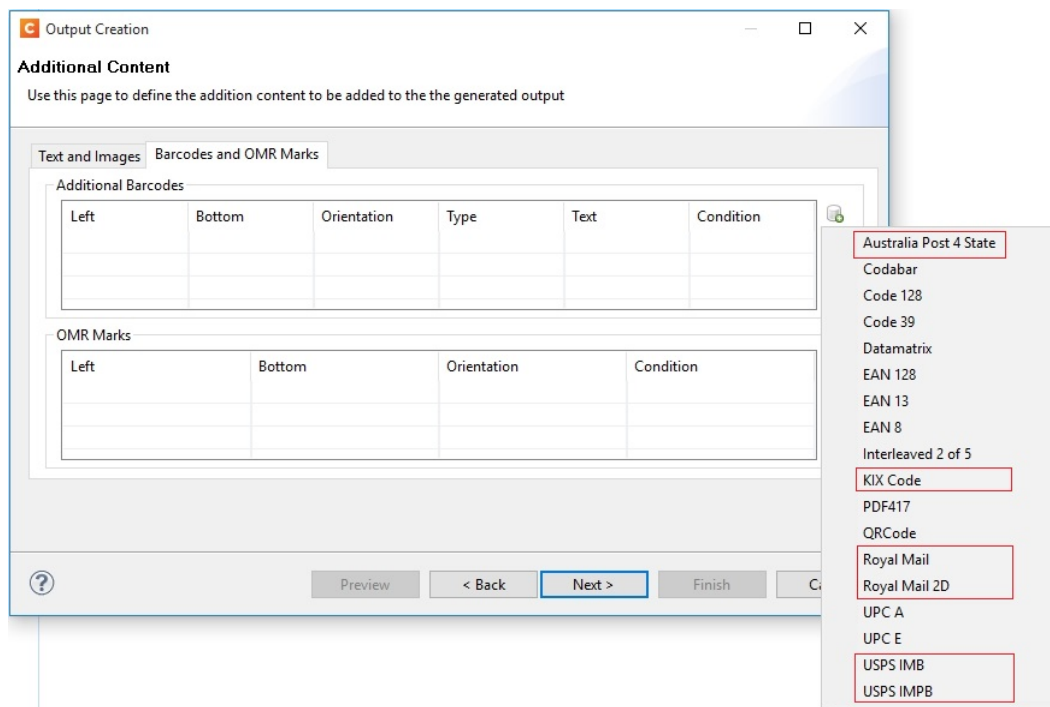
## Additional Postal Services Barcodes added to Output Creation

Barcodes for postal services are excellent candidates for adding during the Output Creation steps, rather than during Content Creation.

Reasons for this include:

- They often cannot be added during Content Creation because they depend on document size (or weight) and on a sort order that is determined during Job Creation.
- They need to go in a fixed position, dependent upon the envelope window, rather than document design.
- It can be desirable to have templates independent from the postal service doing the delivery, in cases where there is a choice between postal services. This makes it relatively easy to switch to whichever service is offering the better rates.

To support these scenarios better, a number of postal service specific barcodes have been added to Output Creation, in addition to Content Creation. (SHARED-54755/54962/55046)



The new barcodes include:

- Australia Post 4 State  

- KIX Code (Dutch postal service - Post.NL)  

- Royal Mail (UK)  

- Royal Mail 2D (UK)  

- USPS IMB (US)  

- USPS IMPB (US)  


Some of these barcodes have specific requirements in order for them to be usable. The respective postal services provide specifications and sometimes also the tools for generating the content of these barcodes.

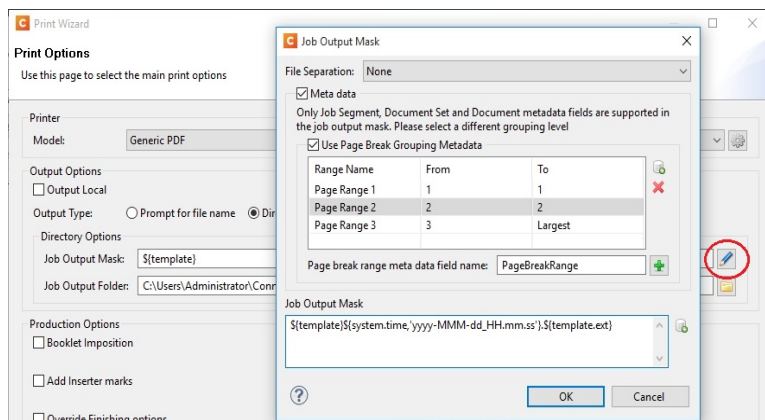
The checksums needed for Australia Post 4 State and IMPB are calculated automatically.

#### Fixed issue with Merge Engine memory usage

The Merge Engine would slow down when running some jobs that used external JavaScript files. These memory issues have now been resolved. (SHARED-47242)

## Job Output Mask improvements, to simplify working with output file names

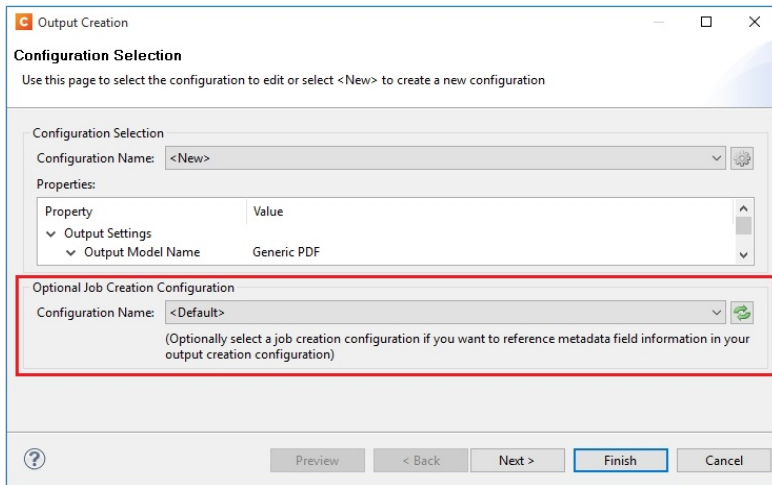
We have improved the way that output file names can be specified. A new dialog box has been added to the Print Wizard, to simplify the creation of Job Output Masks. While it is still possible to directly type a file name with placeholders in the Output File Mask box, it is now also possible to use the dialog to pick the metadata fields and other variables that can be used to create dynamic file names. (SHARED-12173)



A typical use case for using place holders in an Output File Mask is while generating PDF's for archiving purposes. This can require generating one PDF per document and often the files have to be named in a meaningful manner, by using an invoice number in the file name for instance. This requires one to define the invoice number as metadata in a Job Preset and then this metadata field can be used in the Output File Mask of the output preset. In addition, the Separation setting of the Output Preset has to be set to separate at the document level.

So the next time you need a dynamically generated output file name like `inv-${document.metadata.InvNumber}.pdf` OR `${document.metadata.ID}-${system.time, 'yyyy-MMM-dd'}.${template.ext}`, you can use this dialog to help you get what you need.

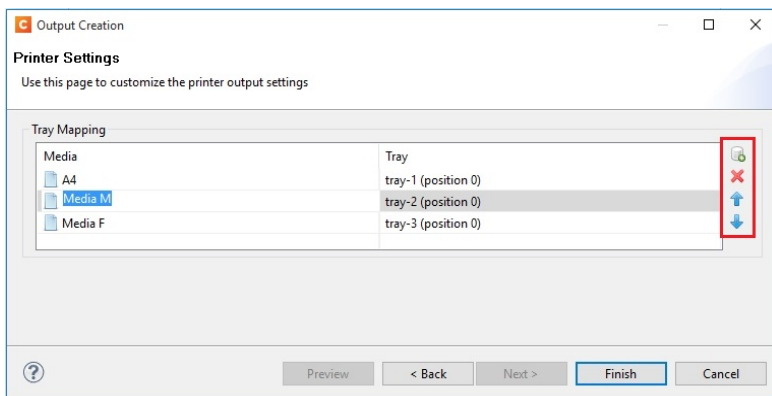
For the Output Preset to know what metadata is available, you can select a Job Preset when creating or modifying an Output Preset:



In the Advanced mode of the Print Wizard this new dialog works a bit different, because the metadata can be directly edited in the same wizard instead of having to refer to a Job Preset.

### Tray Mapping for Multiple Templates

For printing to a cutsheet printer, the Output Preset allows mapping of media defined in a template to trays and media known by the printer. To make it easier to use an Output Preset for multiple templates, the list of media shown on the Tray Mapping page is no longer fixed. So now it is possible to easily define a tray mapping for all media used on a certain printer. (SHARED-49357)



This doesn't mean that all these media have to be used in every job, so one might even map multiple media types to the same tray. In such cases, a Job Preset could be used to filter jobs in

such a way that no conflicting tray mappings can occur within a job, as Job Presets allow filtering by media type.

## Print Output

- **Improvements made to the Print Wizard**

These include:

- Improved usability in Inserter dialog. (SHARED-38279)
- Data Filtering dialog usability improved. (SHARED-38281/38283)
- Support added for manually setting both the horizontal and vertical gutter settings in Booklets (SHARED-53769)
- The Additional Text and Barcode dialogs did not allow many of the available system fonts to be used. This issue has been fixed. (SHARED-46825)
- **Improved PDF comparison** implementation has improved output creation times. (SHARED-44097)
- **PostScript** Tray mapping configurations can now be made independent of the loaded template. (SHARED-49357)
- Improve output creation speed when **outputting with separation**. (SHARED-52088)
- **Soft masked images** were not handled correctly when writing to PDF/VT, causing errors. This issue has now been fixed. (SHARED-32335)

## Workflow 8.7 Enhancements and Fixes

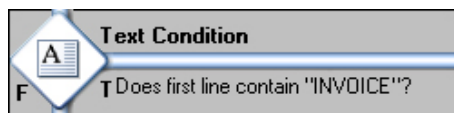
### Custom Task descriptions

The Comments section of each Workflow Task can now be used as the task's description in the Workflow Configuration tool, allowing users to better document the process without having to resort to numerous Comment Tasks. (SHARED-39120)

Workflow processes can sometimes become rather complex and thus they require some documentation in order to allow subsequent users to know why they were implemented in one fashion or another. *Comment tasks* are already available but they use up some valuable real estate in Workflow processes, which sometimes adds to the clutter, rather than making things clearer.

Each task in a process also has a *Comments* section that allows you to properly document what you want, but requires you to click each task in order to view the comments associated with each (and you have to remember to display the *Task Comments* panel, which also robs you of valuable on-screen real estate).

To cure these issues in Workflow 8.7 we introduced a new checkbox located below the Comments field: *Use as step description*. Ticking this box instructs the Configuration tool to use the task's Comments as the description for the task in the Process panel, which allows you to put in more descriptive text than the default value without having to click on each Task to visualize it:



### Option to bypass Record Persistence added to plugins

A new checkbox option has been added to both the **Create PDF Preview** task and the DataMapper tab of the **All-In-One** task, allowing you to specify that the data should not be stored in the database. This feature is specifically tailored for one-off jobs, to prevent data from being written needlessly to the database. Instead, records are streamed directly into the Content Creation process for immediate merging. Turning the feature on in the All-In-Task can improve data mapping performance significantly, as well as the time required for the clean-up process.

Note that checking this option in the All-In-Task means that if you ever need the data for any

reason (reprints, produce additional jobs, etc.), you will have to perform the data mapping configuration from scratch. So make sure you only tick the box for true one-off jobs. (SHARED-48956/56420)

#### Retrieve Items task now has a JSON Output option

The **Retrieve Items** task can now output the results of its query as a JSON string instead of storing them within the metadata. This allows easy handling of the results either through Workflow Scripting or directly in the Designer.

The Retrieve Items task is often used in circular Workflows where data that has already been extracted, output and presented to the user is sent back by the user for further processing. Think, for instance, of a Proof-Of-Delivery Workflow where the delivery person makes the recipient sign a **Capture OnTheGo** document and dynamically makes adjustments to the quantities that were actually shipped out. Both the signature and the modified values are sent back to a Workflow process whose purpose is to generate a PDF version of the document with those modified values.

With the new option to retrieve existing items in JSON format (including the detail tables), you can now retrieve the original record as a JSON object and use a simple script to update all the detail line values using those posted by the end-user. You can then provide that JSON object directly to the Create PDF Preview task without having to perform the data mapping operation once again, thus saving some valuable resources (both in terms of time and database space). (SHARED-50426)

#### New Create Preview PDF plugin added

A new **Create Preview PDF** plugin has been added to the Connect Workflow. The Plugin retrieves the resulting PDF from the file store and makes it available to the process as the job data file. (SHARED-47860)

#### Create Email Content task now has an option to test the SMTP settings

The **Create Email Content** task now has an option to test the SMTP settings entered before saving the configuration. (SHARED-44332/46165)

## Note

The "Test SMTP Settings" does not work when using TLS. This limitation will be addressed in a later release.

### General Workflow fixes and enhancements

- **Improved datamapping speed** when outputting records in metadata. (SHARED-38455)
- **Improved performance** when creating metadata after Content Creation. (SHARED-47150)
- Processing a **Secure PDF as passthrough** through CreatePDF will retain the Security options. (SHARED-47951)
- The **Create Email Content** task now validates the format of the sender's address to prevent typos/mistakes from being saved to the configuration. (SHARED-47947)
- The **OL Connect Send Plug-In** now stores user name and IP along with Job Info. (SHARED-49421)
- The **Create Web Content** task can now be added to processes as an Output task. This helps make processes leaner and easier to understand. (SHARED-50083)
- Drop down list added to the **HTTP Server Input** task to set the application/json mime type. (SHARED-50085)
- **OL Connect Send related plugins** now all grouped together. (SHARED-50126)
- An issue has been fixed where the **wrong document** was attached to email output. This occurred when generating email output with a PDF attachment based on a print context and generating print output of that same template in a single Workflow process. (SHARED-51396)
- The **Output to SharePoint** task was hard coded to use MSXML 4, which was installed along with Workflow. This hard-coded dependency has now be removed from any task that uses MSXML so that they can now automatically use the latest version installed on the PC. (SHARED-53831)



# Known Issues

## Issues with Microsoft Edge browser

The Microsoft Edge browser fails to display web pages when the Workflow's CORS option (in the HTTP Server Input 2 section) is set to "\*". This issue will be resolved in a future release.

## Installation Paths with Multi-Byte Characters

When installing the Chinese (Traditional or Simplified) or Japanese versions of Connect, if the user specifies an alternative installation path containing multi-byte/wide-char characters it can break some of the links to the Connect-related shortcuts in the Start Menu and cause an error to appear at the end of the installer. The workaround for the moment is to use the default installation path. The problem will be addressed in a later release.

## Switching Languages

Changing the language using the **Window>Preferences>Language Setting** menu option does not currently change all of the strings in the application to the selected language. This is a known issue and will be fixed in a later release.

In the meantime we offer the following workaround for anyone who needs to change the language:

1. Go to the .ini files for the Designer and Server Config:
  - C:\Program Files\Objectif Lune\OL Connect\Connect Designer\Designer.ini
  - C:\Program Files\Objectif Lune\OL Connect\Connect Server Configuration\ServerConfig.ini
2. Change the language parameter to the required one under Duser.language=en | es | de | fr | it | ja | pt | tw | zh

Only one of the above language tags should be selected. Once saved, Connect will appear in the selected language at next start-up.

## GoDaddy Certificates

When installing Connect offline, dialogs allow installing the GoDaddy certificates. Most users should use the default settings and click **Next**. In some cases, however, this may not work correctly. For this reason those users should activate **Place all certificates in the following store** and then select the **Trusted Root Certification Authorities** as the target certificate store.

## MySQL Compatibility

After installing Connect 1.7.1 a downgrade to a Connect version earlier than Connect 1.3 or to a MySQL version earlier than 5.6.25 is not seamlessly possible. This is because the database model used in Connect 1.3 and later (MySQL 5.6) is different to that used in earlier versions. If you need to switch to an older version of Connect / MySQL, it is first necessary to remove the Connect MySQL Database folder from "%ProgramData%\Connect\MySQL\data" before installing the older version.

## PostScript Print Presets


The print presets for PostScript were changed from Version 1.1 onwards meaning that some presets created in Version 1.0 or 1.0.1 may no longer work.

Any PostScript print preset from Version 1.0 that contains the following will not work in Version 1.7.1: \*.all[0].\*

Any preset containing this code will need to be recreated in Version 1.7.1.

## Available Printer Models

Note that only the single Printer Model (Generic PDF) will appear on the **Advanced** page of the **Print Wizard** by default.

To add additional printer models click on the settings  button next to the Model selection entry box.

Note that the descriptions of some of the printers were updated in version 1.2 meaning that if you had version 1.n installed, you may find that the same printer style appears twice in the list, but with slightly different descriptions.

For example the following printer types are actually identical:

- Generic PS LEVEL2 (DSC compliant)
- Generic PS LEVEL2 (DSC)

## External Resources in Connect

There are certain limitations on how external resources can be used in Connect. For example if you want to link a file (e.g., CSS, image, JavaScript etc.) from a location on the network but you

do not want to have a copy of the file saved with the template you need to do the following:

1. The resource must be located where it can be accessed by all Servers/Slaves run as users. Failure to do this will cause the image to appear as a Red X in the output for all documents which were merged by engines which could not access the file. The job will terminate normally and the error will be logged.
2. The file must be referenced via a UNC path e.g.,  
file:///w2k8r2envan/z%20images/Picture/Supported/JPG/AB004763.jpg
  - UNC paths are required because the services will be unable to access mapped network drives (Windows security feature).
  - The engine processing the job will look on the local file system for the direct file path leading to the “resource not found” issue mentioned above.

### **Warning**

Important Note: The Designer itself and Proof Print do not use processes that run as services and they may find local files with non-UNC paths which can lead to the false impression that the resources are correct.

### Using Capture After Installing Workflow 8

If PlanetPress Connect Workflow 8 is installed alongside PlanetPress Suite Workflow 7, Capture can no longer be used within Workflow 7. The plugins are now registered uniquely to Workflow 8 and the Messenger for Workflow 7 is taken offline. It is only possible to use Capture from PlanetPress Connect Workflow 8 thereafter.

### Capturing Spool Files After Installing Workflow 8

If PlanetPress Connect Workflow 8 is installed alongside PlanetPress Suite Workflow 7, the PlanetPress Suite 7 option to capture spool files from printer queues will no longer function. The solution is to use PlanetPress Connect Workflow 8 to capture spool files from printer queues.

### Colour Model in Stylesheets

The colour model of colours defined in a stylesheet can sometimes change after editing the stylesheet. This is a known issue and will be addressed in a subsequent release.

## Online Help Links Point to Introductory Page

Context sensitivity for the online help is not yet enabled in Connect. All links and F1 calls point to the introductory page, where you can Search on keywords to bring up Help pages relating to the topic.

Context sensitivity will be introduced in a subsequent release of Connect.

## Image Preview in Designer

If in the Windows Internet settings (**Connection Settings > LAN configuration**) a proxy is enabled, but "Bypass proxy settings for local addresses" is not checked, the image preview service, conversion service and live preview tab in the Designer will not work and exhibit the following issues:

- Images will be shown as 0 size boxes (no red 'X' is displayed)
- Live preview does not progress, and when re-activated reports "browsers is busy"

To fix the issue you must check the "Bypass proxy settings for local addresses" option.

## MergeWeaver Engines when Printing

The print operation in the Designer will automatically detect whether the Merge\Weaver engines are available and display a message for the user to retry or cancel if not. Once the MergeWeaver engine becomes available and the user presses retry the print operation will proceed as normal. This message can also occur in the following circumstances:

- If the server is offline and you are not using Proof Print
- On some occasions before the Print Wizard opens

## REST Calls for Remote Services

The Server will now accept REST calls for all remote services and will make commands wait indefinitely until the required engines become available. The Server will log when it is waiting for an engine and when it becomes available. Note that there is no way to cancel any commands other than stopping the Server.

## Print Content and Email Content in PlanetPress Workflow

In PlanetPress Workflow's Print Content and Email Content tasks, the option to Update Records from Metadata will only work for fields whose data type is set to String in the data

model. Fields of other types will not be updated in the database and no error will be raised. This will be fixed in a later release.

### Print Limitations when the Output Server is located on a different machine

The following limitation may occur when using the Print options from a Designer located on a different machine to the Output Server:

- The file path for the prompt and directory output modes is evaluated on both the client AND server side. When printing to a network share it must be available to BOTH the Designer and Server for the job to terminate successfully.
- The Windows printer must be installed on both the Server and Designer machines.
- When printing via the Server from a remote Designer, the output file remains on the Server machine. This is remedied by selecting "Output Local" in the Output Creation configuration.

### VIPP Output

Some templates set up with landscape orientation are being produced as portrait in VIPP. It can also sometimes be the case that text and images can be slightly displaced. These are known issues and will be addressed in a later release of Connect.

## Overview

This document provides an overview of the new features and enhancements in PlanetPress Connect 1.6.1 and PlanetPress Workflow 8.6.

The major focus of Connect 1.6 has been to improve performance and increase stability, as well as launching a new option for the PReS Connect and PlanetPress Connect brands called **OL Connect Send**.

A description of the the new **OL Connect Send** can be found here: "OL Connect Send" on page 1052.

### Installing PlanetPress Connect 1.6.1 and PlanetPress Workflow 8.6

- PlanetPress Connect is released as a 64 Bit version only (with the exception of the Workflow, Fax, Search and Imaging modules).
- Full details on installing and licensing PlanetPress Connect and PlanetPress Workflow can be found in the online help in the installer.

- Note that both PlanetPress and PlanetPress Connect Workflow come with a 30 day trial licenses by default.

### Updating from PlanetPress Connect 1.1

In order to upgrade from Connect Version 1.1 to Version 1.6.1 via the Update Manager it is necessary to install a new version of the Objectif Lune Update Client. The next time you run your current Update Client it will show that there is an update available for itself. Simply click on the download button in the dialog to install the new version of the Update Client. Note that it is no problem to run the update while the Client is open. It will automatically update itself.

Once you have done this, PlanetPress Connect 1.6.1 will become available for download.

From Connect Version 1.2.0 onwards, the newer version of the Update Client was included with the Connect installation.

### Updating stand-alone Workflow Messenger installations

If Workflow Messenger were installed stand alone, with no other Workflow components installed, the Update Client cannot find the Messenger component and thus it will not automatically update the component to the Workflow 8.6 version of Messenger. To get around this, download and run the Workflow 8.6 installer manually.

### Templates Used in Workflow

For improved performance we recommend re-saving Workflow templates set up in the previous versions to run with PlanetPress Connect 1.6.1\Workflow 8.6.

### Updating Connect 1.5 installations using Microsoft SQL Server as back-end

If a Microsoft Server and MySQL were installed with Connect 1.5, and the Server Configuration Tool used to switch the back-end database to Microsoft SQL, then an extra step is required in the Update to Connect 1.6.1. The procedure is documented in the installation guide.

### Print Only Version

A Print Only license is available with version 1.6.1 of PlanetPress Connect which allows legacy PlanetPress Suite 7 customers on OL Care to upgrade to Connect for a minimal fee. The license allows regular printing via the Print Wizard but runs Email and Web output in demo mode. For more information, please contact your local OL Customer Care or Sales team.

## Reduced Memory Version

### **Note**

This is **not** recommended for production.

It is now possible to install PlanetPress Connect on a machine with a minimum of 2 GB of RAM. The PlanetPress Connect Designer will automatically detect whether it has been installed on a machine with less than 4 GB of RAM and default to only using one internal Weaver and one internal merge engine on that system. The Server will also run using internal engines.

# OL Connect Send

OL Connect Send is an application of two components. The first is a Windows printer driver and the second is a set of Workflow plug-ins.

In its most basic form, OL Connect Send allows the transmission of print files over the Internet from any Windows Desktop application.

## OL Connect Send flavors

OL Connect Send comes in three flavors. These are:

- **Free of charge:** No license required; any user; any domain; no usage limits; no web interaction.
- **User mode:** user-domain license required for each client domain; no usage limits, web interaction. OL Care at POP sold separately. Packages are available for 10, 25, 50, 100, 200 or Enterprise (Unlimited) users.
- **Credit mode:** credit license required; any user; any client (allows all domains) ; limited by credits only, web interaction. Includes OL Care at POP. Credits packages available for 20,000, 50,000, 100,000, 200,000 credits, 400,000 and 1,000,000 credits.

With the licensed version, OL Connect Send has the ability to requests a web page, displayed in the user's browser that allows them to enter job specific information. The information from this web page tells Workflow what to do next. OL Connect Send can be used to create custom interactive workflows from a centralized location, yet has the ability to be deployed and installed very easily.

## OL Connect Send Verticals and applications

OL Connect Send has many applications. These include:

- **Print-for-Pay market:** Enables the consolidation of incoming jobs for further processing, printing and mailing.
- **Insurances and Financial market:** Desktop capture over the Internet of print jobs, (remote workers and branches), for centralized processing, printing and mail.
- **Law firms:** Desktop capture over the Internet of print jobs requiring complex mail merge and stationery management.



- **Supply Chain:** Inbound document processing, such as capturing inbound invoices or POs for publication in an ECM. Print to EDI for outbound documents such as invoices.

For further information on Connect Send, please refer to the OL Connect Send website and standalone User Guide.

# Connect 1.6.1 General Enhancements and Fixes

## Performance improvements

- Changes to the handling of **transparency in PDF backgrounds** has not only cured some job failures, but has also led to substantial improvements to both output speeds and filesizes. (49680)
- Improved processing speed for **multiple large detail table** documents. (47252/48537)
- Improvements made to the **clean-up** processes, improving overall production speed.
- Some **memory leaks plugged**, improving overall production speed.
- Improved reliability when using **MS-SQL** as back-end database.

Further Performance improvements can be found detailed in the [Output Enhancements and Fixes](#) section.

## Documentation improvements

- Help layout changed to allow **easier navigation**, and the content improved.
- Broken links within the **Welcome Screen** have been fixed. (39077/47964)

## Installer improvements

- **Microsoft SQL Server** can now be setup as the back-end database during the installation process. (47546)

# Connect 1.6.1 Designer Enhancements and Fixes

## General Designer improvements

- **Interface improvements** such as inclusion of icons for different types of files (js and CSS).
- Provided option to configure the **script timeout period**. (48639)
- Minor issues with non-English language **translations** fixed.
- Display issues that were sometimes encountered when changing **section background images** have been fixed.
- Issue with Virtual Stationery not being enabled when adding VS PDF in **Formal Letter Template** wizard now fixed. (47268)
- Fixed an issue in which **resizing an abs box** in some scenarios would result in an additional box being displayed. (47858)
- When **typing in a non-English input language** (such as Hebrew), the keyboard input would periodically switch to English. This issue has been fixed. (49566)
- Saving a template when in Source view would erroneously **show snippet content** within the Source view thereafter. This has been fixed. (50131)
- HEX encoding option added to **DataMatrix** barcodes. (47897)

## Capture On The Go (COTG) and Web form improvements

- **Dummy data** can now be added, to help when designing COTG forms. (47835)
- Improvements made to the COTG **Kitchen Sink** template Wizard. Added Image Annotation and Fields Table controls. (47817)
- Visible container added to **Geolocation** objects.
- Added support for **blank forms**, for multiple submissions. (50483)
- Workflow "**Output to COTG**" task now returns Document ID. (50487)
- Fixed picture issues with **iOS** sometimes causing image to be displayed as a red "X". (45231)
- Fixed Geolocation coordinates on **Android** devices. (46831)

## Connect 1.6.1 DataMapping Enhancements and Fixes

- Support for **Regular Expressions** added to database searches. (51694)
- Improved Datamapping process **reliability**.
- Improved data record reliability when handling **large jobs** (those in excess of 50,000 records).
- Improved **PDF extraction** avoids character duplication.
- Improved **marking of data fields** in extraction steps.
- New option added to **support multibyte** (variable length) encoded data such as Big5, GBK, UTF-8 and Shift-JIS. (46813)
- Fixed issue whereby post-function for **extracted fields containing several lines** would only replace characters on the first line. (47949)
- Fixed issue with **Identically named datafields** in different detail tables causing issues with data extraction.
- Fixed issues with **Right-To-Left text concatenation**. (48712)
- Fixed issues with **PlanetPress Suite PDFs** not working consistently within Connect. (50355)

## Connect 1.6.1 Output Enhancements and Fixes

### General

- Merge Engine **memory leaks** fixed. (50188)
- Improved creation of **Metadata** after content creation.
- Improved conditional content for **Email and Web output**.
- Fixed issue with PDF Pass-through jobs, whereby **hidden features of the PDF** input were being included in the output. (49373)
- Some templates containing JPG or Bitmap graphics could trigger the **flatten.exe** program (used for flattening external files) to repeatedly open, causing performance degradation or even failure. (51251)

## Email Output

- Fixed issue with the **email Subject field** not being encoded properly when using characters other than Latin characters. (48781)
- To **improve privacy** certain Meta tags that were embedded in the output email HTML have been removed. These include "email-reply-to", "email-from", "sender-name" and "sender-address". (49864)
- The **Date field** was not always included in the email header. This has been fixed and the Date field should now be present in all email headers. (48706)

## Print Output

- Improved processing speed for jobs involving **separation** that create multiple output files. (49167)
- Improvements made in the **conversion of PDFs** to other formats have made such conversions significantly faster and the output smaller, in PCL output. (50140)
- Improved **page range handling** on last page.
- Improved **N-Up** positioning.
- If an image file were replaced on disk between runs then a subsequent **Proof printing** would display the old image, . This has been fixed. (47567)
- Merge engine no longer slows down when using **external JavaScript** files in print sections. (48447)
- Fixed "**ApplicationException: Null**" errors encountered in some PCL outputs. (50868)

# Connect Workflow 8.6 Enhancements and Fixes

## General improvements

- A multitude of changes and enhancements made to support new **OL Connect Send** functionality within Workflow.
- Support for **password protection** added to "Create PDF" task. (48380)
- **Generic Data Repository** field length extended beyond previous limit of 32 characters. (47734)
- Added new "**Create PDF Preview**" task for lightning fast creation of single record PDF. (49497)
- Minor issues with some **language translations** fixed.
- Fixed issue with **control script tags** not being set correctly in email headers, in emails sent via Workflow. (51708)
- Fixed issue with the HTTP Server Input plug-in not receiving any data when a **webform contains the `enctype="multipart/form-data"` instruction** and no files were attached to the form. (35751)
- When running a process that produces an email with a PDF attachment along with a print output in the same process, the **email attachment would be incorrect** (that of the previous run). This issue has been fixed. (51879)

## Performance and Stability improvements

- Improved **Print Content Creation** speed in both stand-alone and "All-In-One" tasks. (49589)
- Improved reliability in **simultaneous (multi-threaded)** COTG uploading. (47146)
- Option added to "**All-In-One**" plug-in to disable the storing of temporary data if not needed.  
This speeds up both output and the cleanup afterward. (50713)
- Fixed issues with **inconsistent PDF output** when calling the *Get Result* method (REST API) in the *All-In-One* process. (49413)
- Fixed issue with Connect (Merge Engines) losing performance when **large numbers of smaller jobs** were processed through Workflow. (50863)

- Fixed potential out-of-memory error with **very large Workflow configuration** files. (51621)
- Fixed crashes encountered when **large numbers of data selection calls** were issued in highly threaded processes. (50569)

#### HTTP and SMTP Server improvements

- Support added for **cross-origin HTTP** (CORS HTTP) requests, to facilitate the development and testing of web templates. (47014)
- Added option to **specify SMTP port number** in "Create Email Content" task. (49887)
- Documentation on **how to retrieve attachments from SMTP input** improved. (49473)
- Fixed corruption of **large uploaded files**. (47901)
- Fixed the timeouts encountered when the option "**Do not include XML envelope**" were selected. (49186)
- Fixed the **HTTP 501 error** when receiving Authorization headers other than Basic. (48387)

#### Capture On The Go (COTG) improvements

- "*Output to COTG*" task now returns **Document ID**. (50487)
- Fixed issue with "*Output to COTG*" task that could sometimes cause memory corruption, requiring a restart of the services. (47804)

# Known Issues

## Installing OL Connect Send on a machine with Connect installed.

When OL Connect Send Plug-Ins are installed (either standalone or via a Workflow installation) on the same machine as Connect, an interference between OL Connect Send's internal Database and that of Connect may occur, which will block a browser popup on that same machine.

This issue can be fixed by applying a startup wait to the Connect Server Service. If the issue occurs during runtime, restarting the Connect Server Service will fix the issue.

## OL Connect Send issues under Microsoft Edge browser.

- Issues with using the built-in Windows 10 default "Administrator" account and the OL Connect Send Client. This is due to Windows disallowing the opening an Edge browser whilst running under this account. This blocks interaction with the printer driver.
- The Microsoft Edge browser fails to display web pages when Workflow's CORS option (in the HTTP Server Input 2 section) is set to "\*". This issue will be resolved in a future release.

## Installation Paths with Multi-Byte Characters

When installing the Chinese (Traditional or Simplified) or Japanese versions of Connect, if the user specifies an alternative installation path containing multi-byte/wide-char characters it can break some of the links to the Connect-related shortcuts in the Start Menu and cause an error to appear at the end of the installer. The workaround for the moment is to use the default installation path. The problem will be addressed in a later release.

## Switching Languages

Changing the language using the **Window>Preferences>Language Setting** menu option does not currently change all of the strings in the application to the selected language. This is a known issue and will be fixed in a later release.

In the meantime we offer the following workaround for anyone who needs to change the language:



1. Go to the .ini files for the Designer and Server Config:
  - C:\Program Files\Objectif Lune\OL Connect\Connect Designer\Designer.ini
  - C:\Program Files\Objectif Lune\OL Connect\Connect Server Configuration\ServerConfig.ini
2. Change the language parameter to the required one under Duser.language=en | es | de | fr | it | ja | pt | tw | zh

Only one of the above language tags should be selected. Once saved, Connect will appear in the selected language at next start-up.

### GoDaddy Certificates

When installing Connect offline, dialogs allow installing the GoDaddy certificates. Most users should use the default settings and click **Next**. In some cases, however, this may not work correctly. For this reason those users should activate **Place all certificates in the following store** and then select the **Trusted Root Certification Authorities** as the target certificate store.

### MySQL Compatibility

After installing Connect 1.6.1 a downgrade to a Connect version earlier than Connect 1.3 or to a MySQL version earlier than 5.6.25 is not seamlessly possible. This is because the database model used in Connect 1.3 and later (MySQL 5.6) is different to that used in earlier versions. If you need to switch to an older version of Connect / MySQL, it is first necessary to remove the Connect MySQL Database folder from "%ProgramData%\Connect\MySQL\data" before installing the older version.

### PostScript Print Presets


The print presets for PostScript were changed from Version 1.1 onwards meaning that some presets created in Version 1.0 or 1.0.1 may no longer work.

Any PostScript print preset from Version 1.0 that contains the following will not work in Version 1.6.1: \*.all[0].\*

Any preset containing this code will need to be recreated in Version 1.6.1.

### Available Printer Models

Note that only the single Printer Model (Generic PDF) will appear on the **Advanced** page of the **Print Wizard** by default.

To add additional printer models click on the settings  button next to the Model selection entry box.

Note that the descriptions of some of the printers were updated in version 1.2 meaning that if you had version 1.n installed, you may find that the same printer style appears twice in the list, but with slightly different descriptions.

For example the following printer types are actually identical:

- Generic PS LEVEL2 (DSC compliant)
- Generic PS LEVEL2 (DSC)

### External Resources in Connect

There are certain limitations on how external resources can be used in Connect. For example if you want to link a file (e.g., CSS, image, JavaScript etc.) from a location on the network but you do not want to have a copy of the file saved with the template you need to do the following:

1. The resource must be located where it can be accessed by all Servers/Slaves run as users. Failure to do this will cause the image to appear as a Red X in the output for all documents which were merged by engines which could not access the file. The job will terminate normally and the error will be logged.
2. The file must be referenced via a UNC path e.g.,  
file:///w2k8r2envan/z%20images/Picture/Supported/JPG/AB004763.jpg
  - UNC paths are required because the services will be unable to access mapped network drives (Windows security feature).
  - The engine processing the job will look on the local file system for the direct file path leading to the “resource not found” issue mentioned above.

#### **Warning**

Important Note: The Designer itself and Proof Print do not use processes that run as services and they may find local files with non-UNC paths which can lead to the false impression that the resources are correct.

### Using Capture After Installing Workflow 8

If PlanetPress Connect Workflow 8 is installed alongside PlanetPress Suite Workflow 7, Capture can no longer be used within Workflow 7. The plugins are now registered uniquely to Workflow 8 and the messenger for Workflow 7 is taken offline. It is only possible to use Capture from PlanetPress Connect Workflow 8 thereafter.

### Capturing Spool Files After Installing Workflow 8

If PlanetPress Connect Workflow 8 is installed alongside PlanetPress Suite Workflow 7, the PlanetPress Suite 7 option to capture spool files from printer queues will no longer function. The solution is to use PlanetPress Connect Workflow 8 to capture spool files from printer queues.

### Colour Model in Stylesheets

The colour model of colours defined in a stylesheet can sometimes change after editing the stylesheet. This is a known issue and will be addressed in a subsequent release.

### Online Help Links Point to Introductory Page

Context sensitivity for the online help is not yet enabled in Connect. All links and F1 calls point to the introductory page, where you can Search on keywords to bring up Help pages relating to the topic.

Context sensitivity will be introduced in a subsequent release of Connect.

### Image Preview in Designer

If in the Windows Internet settings (**Connection Settings > LAN configuration**) a proxy is enabled, but "Bypass proxy settings for local addresses" is not checked, the image preview service, conversion service and live preview tab in the Designer will not work and exhibit the following issues:

- Images will be shown as 0 size boxes (no red 'X' is displayed)
- Live preview does not progress, and when re-activated reports "browsers is busy"

To fix the issue you must check the "Bypass proxy settings for local addresses" option.

## MergeWeaver Engines when Printing

The print operation in the Designer will automatically detect whether the MergeWeaver engines are available and display a message for the user to retry or cancel if not. Once the MergeWeaver engine becomes available and the user presses retry the print operation will proceed as normal. This message can also occur in the following circumstances:

- If the server is offline and you are not using Proof Print
- On some occasions before the Print Wizard opens

## REST Calls for Remote Services

The Server will now accept REST calls for all remote services and will make commands wait indefinitely until the required engines become available. The Server will log when it is waiting for an engine and when it becomes available. Note that there is no way to cancel any commands other than stopping the Server.

## Print Content and Email Content in PlanetPress Workflow

In PlanetPress Workflow's Print Content and Email Content tasks, the option to Update Records from Metadata will only work for fields whose data type is set to String in the data model. Fields of other types will not be updated in the database and no error will be raised. This will be fixed in a later release.

## VIPP Output

Some templates set up with landscape orientation are being produced as portrait in VIPP. It can also sometimes be the case that text and images can be slightly displaced. These are known issues and will be addressed in a later release of Connect.

## Print Limitations when the Output Server is located on a different machine

The following limitation may occur when using the Print options from a Designer located on a different machine to the Output Server:

- The file path for the prompt and directory output modes is evaluated on both the client AND server side. When printing to a network share it must be available to BOTH the Designer and Server for the job to terminate successfully.
- The Windows printer must be installed on both the Server and Designer machines.

- When printing via the Server from a remote Designer, the output file remains on the Server machine. This is remedied by selecting “Output Local” in the Output Creation configuration.

## Overview

This document provides an overview of the new features and enhancements in PlanetPress Connect 1.5 and PlanetPress Workflow 8.5.

### Installing PlanetPress Connect 1.5 and PlanetPress Workflow 8.5

- PlanetPress Connect is released as a 64 Bit version only (with the exception of the Workflow, Fax, Search and Imaging modules).
- Full details on installing and licensing PlanetPress Connect and PlanetPress Workflow can be found in the online help in the installer.
- Note that both PlanetPress and PlanetPress Connect Workflow come with 30 day trial licenses by default.

### Updating from PlanetPress Connect 1.1

In order to upgrade from Connect Version 1.1 to Version 1.5 via the Update Manager it is necessary to install a new version of the Objectif Lune Update Client. The next time you run your current Update Client it will show that there is an update available for itself. Simply click on the download button in the dialog to install the new version of the Update Client. Note that it is no problem to run the update while the Client is open. It will automatically update itself.

Once you have done this, PlanetPress Connect 1.5 will become available for download.

From Connect Version 1.2.0 onwards, the newer version of the Update Client was included with the Connect installation.

### Updating stand-alone Workflow Messenger installations

If Workflow Messenger were installed stand alone, with no other Workflow components installed, the Update Client cannot find the Messenger component and thus it will not automatically update the component to the Workflow 8.5 version of Messenger. To get around this, download and run the Workflow 8.5 installer manually.

## Print Only Version

A Print Only license is available with version 1.5 of PlanetPress Connect which allows legacy PlanetPress Suite 7 customers on OL Care to upgrade to Connect for a minimal fee. The license allows regular printing via the Print Wizard but runs Email and Web output in demo mode. For more information, please contact your local OL Customer Care or Sales team.

## Templates Used in Workflow

For improved performance we recommend re-saving Workflow templates set up in the previous versions to run with PlanetPress Connect 1.5\Workflow 8.5.

## Reduced Memory Version

### Note

This is **not** recommended for production.

It is now possible to install PlanetPress Connect on a machine with a minimum of 2 GB of RAM. The PlanetPress Connect Designer will automatically detect whether it has been installed on a machine with less than 4 GB of RAM and default to only using one internal Weaver and one internal merge engine on that system. The Server will also run using internal engines.

# Connect 1.5 Designer Enhancements and Fixes

## General Designer improvements

- A **color selection eyedropper** has been added, to allow the selection of a color from elsewhere on screen. (SHARED-33561/33646/36293)
- **Improved responsiveness** within the Designer, particularly when dealing with large and complex documents. (SHARED-44309)
- A **configurable Auto Save functionality** has been added for both templates and DataMapping configurations. (SHARED-40942/42085)
- Improvements made to **image file selection** functionality. (SHARED-42231/42451/42503/42556//43778)

## Simplified creation of templates based on existing PDFs

- Option added to allow the creation of a **new print document based on an existing PDF**. (SHARED-19220)
- Improved support for **adding PDF files as Section backgrounds**. Files can now be referenced from disk or imported into Template. (SHARED-42496)
- Added support for drag and dropping **Data Fields** directly onto the page as absolutely positioned textboxes. (SHARED-43311)

## Print Layout improvements

- **Page Number** formatting options (start/stop page numbering for sections, set numbering notation) improved in Print Section Properties dialog. (SHARED-39048)
- Added **repeating background images** support for print documents. (SHARED-43201)
- Option added to allow the insertion of **absolute positioned tables** on a master page. (SHARED-21967)

## Email enhancements

- **User-definable SMTP settings**. New defaults are added for Sendgrid and Mailgun (in addition to Mandrill). (SHARED-43897)
- The standard **New email** wizard has been replaced with the new **Basic Email template** wizard. The new wizard has improved HTML structure. (SHARED-43338)
- Sending a **test email** no longer requires data. (SHARED-41889)
- **Tighter compression** for PDF attachments that are based on a print section. (SHARED-38575)
- **Colour picker** support added to the Email template wizards. (SHARED-33561)
- Added support for **PNG barcode** images in email messages. (SHARED-43787)

## Barcode enhancements

- **Improved Barcode creation** with improved dialogs, better data validation and better error messaging. (SHARED-39295/42879)
- Font controls added to the **Barcode Properties** dialogs. (SHARED-22722/43659)

- Barcode improvements made in **Preview** mode. Support added for resizing and dragging of absolute positioned barcode objects, as well as resizing of inline barcode objects. (SHARED-43641)
- Barcodes can now have **transparent backgrounds**. (SHARED-43659)

### Scripting improvements

- New **closest()** command added to the Scripting API, to locate closest matching element above it in the Document Object Model (DOM) tree. (SHARED-41789)
- **Script editing improved**. Line numbering now available within the editor, support for code completion and syntax highlighting added, as well as support for various ECMA6 commands. (SHARED-42768/43696)
- Support added for **cloning Sections in a Control Script** to allow a document to have a dynamic number of Sections. (SHARED-43683)
- Improved **Scripts tool tip** warning and error messaging. (SHARED-42550/43758)  
Improvements include:
  - Better tailored error messages and warnings.
  - Icons added representing script type as well as showing the issue severity.
  - Duplicated problems now filtered out.
  - Several other minor improvements.
- Improved support for raw HTML within Designer scripting API commands. (SHARED-43075)

### Capture On The Go (COTG) and Web form improvements

- Input fields residing in a Field Table or Dynamic table are submitted using an array notation (requires Workflow 8.5). This results in a **nested XML structure** (grouped fields) in the job data file, simplifying the Workflow process and extracting data in the DataMapper. (SHARED-45577)
- COTG/Web-Form: Option introduced to retrieve the **jobdata XML** file from within the Designer. An icon has been added to the toolbar that intercepts the Submit action in Live view and then submits the form to a local or remote Workflow engine. A dialog also allows for saving the data file. (SHARED-44899)
- Deploying a COTG **Test form** no longer requires data. (SHARED-41889)



- **New scripting options** have been added to the COTG.js library to register custom functions for save and restore. (SHARED-40670)
- **Colour picker** support has been added to the **COTG Starter Template** wizards. (SHARED-33561)
- Improved speed/size of **COTG Camera** objects. Rather than embedding images in output PDFs, Connect now supports embedding data URLs in COTG templates. (SHARED-38575)
- Documents from the Library now **automatically deleted** upon successful submission. (COTG-367)

## Connect 1.5 DataMapping Enhancements and Fixes

- **Multiple Conditions** step can now evaluate several conditions and branch out accordingly. (SHARED-14329/44435)
- **Performance improvements** made when extracting text from PDFs and spool files. (SHARED-43056)
- **Improved default formatting** when extracting Date, Float or Currency data fields. (SHARED-43415)
- An **extra field** is now appended to every Document record and to every Detail table inside that record. This allows other processes to add data on the fly. This provides enormous flexibility. For example, adding a JSON object (which could contain several additional fields) to the new field value extends the data structure almost infinitely. (SHARED-43518)
- The **JavaScript API** now displays detailed hints for every command, object and method available. (SHARED-44838)

## Connect 1.5 Output Enhancements and Fixes

### General

- Improved **content creation** processing speed for templates featuring PDF backgrounds. (SHARED-44350)

### Email Output

- **Basic Email Action** wizard now made the default for new Email templates. (SHARED-43338)
- Support added for **user defined** SMTP/Email Service Provider (ESP) settings. (SHARED-43897)

### Print Output

- New option added, allowing printing to **Windows Printer Driver**. (SHARED-35536)
- Improvements made to **external sort** option in Job presets. Support added for using input/output file placeholders. (SHARED-40944)

- New **HCF** file added that supports “top down wrap around sequence marks”. (SHARED-42326)
- Use **PostScript Media** name values in the PostScript DSC comments, to improve subsequent searches. (SHARED-42826)
- Option added to allow **storing of job resources** on PostScript printer’s own storage medium. (SHARED-43467)
- **OMR marks improved**, with support added for Match Numbers (Match Code, MC). (SHARED-43589)
- A **Proof preview function** has been added to the Output Wizard, to display onscreen how the current print job would appear when printed. (SHARED-43885)
- **Imposition improvement**. Can now set specific starting position via new Offset option. (SHARED-44022)
- Minor glitches in **Booklet** and **Imposition** output addressed. (SHARED-44340/44430)

#### Web Output

- Extra customization added to **custom OMR settings**. (SHARED-36267)

# Connect 1.5 General Enhancements and Fixes

## Installer improvements

- Improvements made to installation robustness. The installer now copes better when encountered **permissions issues** during installation. (SHARED-43732/43737)
- The **Update Client** has been updated to 1.1.9 and has been included in both the Connect 1.5 and Workflow 8.5 installations. (SHARED-47065)

## Connect 8.5 Workflow Enhancements and Fixes

- Support for PHP-like arrays for **COTG** or web-based form submissions. (SHARED-41706)
- New Workflow system variable (%r) added to allow a process to determine which if it is currently running in **service** or **debug** mode. (SHARED-43411)
- **Create Output** and the Connect **All In One** tasks can now be added as Output tasks without waiting for the operation's result. (SHARED-43413)
- The **Folder Capture** task can now monitor multiple folders. (SHARED-43417)
- The **HTTP Server Input** task can now be set to monitor multiple actions to receive files from different URLs. (SHARED-43419)
- The **Create Web** and **Create Email Content** tasks have been enhanced with a list of the available template sections made available to simplify selection. (SHARED-43421)
- A new **Data Repository** has been created to allow for storing data that can then be subsequently reused, modified or augmented, by different processes.  
A new **Data Repository Management Tool (DRMT)** has also been added to the Workflow, to provide simple repository management tasks. (SHARED-43423/43438/43488/43521)
- Support added for Regular Expressions in the **Folder Capture** task FileName masks. (SHARED-43436)
- The **Debug information** panel is now automatically made visible when debugging a process. (SHARED-43763)
- Additional **encryption options** (RC4 and AES-256 in addition to AES-128) added for password protecting PDF files. (SHARED-44208)

# Known Issues

## Installation Paths with Multi-Byte Characters

When installing the Chinese (Traditional or Simplified) or Japanese versions of Connect, if the user specifies an alternative installation path containing multi-byte/wide-char characters it can break some of the links to the Connect-related shortcuts in the Start Menu and cause an error to appear at the end of the installer. The workaround for the moment is to use the default installation path. The problem will be addressed in a later release.

## Switching Languages

Changing the language using the **Window>Preferences>Language Setting** menu option does not currently change all of the strings in the application to the selected language. This is a known issue and will be fixed in a later release.

In the meantime we offer the following workaround for anyone who needs to change the language:

1. Go to the .ini files for the Designer and Server Config:
  - C:\Program Files\Objectif Lune\OL Connect\Connect Designer\Designer.ini
  - C:\Program Files\Objectif Lune\OL Connect\Connect Server Configuration\ServerConfig.ini
2. Change the language parameter to the required one under Duser.language=en | es | de | fr | it | ja | pt | tw | zh

Only one of the above language tags should be selected. Once saved, Connect will appear in the selected language at next start-up.

## GoDaddy Certificates

When installing Connect offline, dialogs allow installing the GoDaddy certificates. Most users should use the default settings and click **Next**. In some cases, however, this may not work correctly. For this reason those users should activate **Place all certificates in the following store** and then select the **Trusted Root Certification Authorities** as the target certificate store.

## MySQL Compatibility

After installing Connect 1.5 a downgrade to a Connect version earlier than Connect 1.3 or to a MySQL version earlier than 5.6.25 is not seamlessly possible. This is because the database

model used in Connect 1.3 and later (MySQL 5.6) is different to that used in earlier versions. If you need to switch to an older version of Connect / MySQL, it is first necessary to remove the Connect MySQL Database folder from "%ProgramData%\Connect\MySQL\data" before installing the older version.

### PostScript Print Presets


The print presets for PostScript were changed from Version 1.1 onwards meaning that some presets created in Version 1.0 or 1.0.1 may no longer work.

Any PostScript print preset from Version 1.0 that contains the following will not work in Version 1.5: \*.all[0].\*

Any preset containing this code will need to be recreated in Version 1.5.

### Available Printer Models

Note that only the single Printer Model (Generic PDF) will appear on the **Advanced** page of the **Print Wizard** by default.

To add additional printer models click on the settings  button next to the Model selection entry box.

Note that the descriptions of some of the printers were updated in version 1.2 meaning that if you had version 1.n installed, you may find that the same printer style appears twice in the list, but with slightly different descriptions.

For example the following printer types are actually identical:

- Generic PS LEVEL2 (DSC compliant)
- Generic PS LEVEL2 (DSC)

### External Resources in Connect

There are certain limitations on how external resources can be used in Connect. For example if you want to link a file (e.g., CSS, image, JavaScript etc.) from a location on the network but you do not want to have a copy of the file saved with the template you need to do the following:

1. The resource must be located where it can be accessed by all Servers/Slaves run as users. Failure to do this will cause the image to appear as a Red X in the output for all

documents which were merged by engines which could not access the file. The job will terminate normally and the error will be logged.

2. The file must be referenced via a UNC path e.g.,  
file:///w2k8r2envan/z%20images/Picture/Supported/JPG/AB004763.jpg
  - UNC paths are required because the services will be unable to access mapped network drives (Windows security feature).
  - The engine processing the job will look on the local file system for the direct file path leading to the “resource not found” issue mentioned above.

### **Warning**

Important Note: The Designer itself and Proof Print do not use processes that run as services and they may find local files with non-UNC paths which can lead to the false impression that the resources are correct.

### Using Capture After Installing Workflow 8

If PlanetPress Connect Workflow 8 is installed alongside PlanetPress Suite Workflow 7, Capture can no longer be used within Workflow 7. The plugins are now registered uniquely to Workflow 8 and the messenger for Workflow 7 is taken offline. It is only possible to use Capture from PlanetPress Connect Workflow 8 thereafter.

### Capturing Spool Files After Installing Workflow 8

If PlanetPress Connect Workflow 8 is installed alongside PlanetPress Suite Workflow 7, the PlanetPress Suite 7 option to capture spool files from printer queues will no longer function. The solution is to use PlanetPress Connect Workflow 8 to capture spool files from printer queues.

### Colour Model in Stylesheets

The colour model of colours defined in a stylesheet can sometimes change after editing the stylesheet. This is a known issue and will be addressed in a subsequent release.

### Online Help Links Point to Introductory Page

Context sensitivity for the online help is not yet enabled in Connect. All links and F1 calls point to the introductory page, where you can Search on keywords to bring up Help pages relating to the topic.



Context sensitivity will be introduced in a subsequent release of Connect.

### Image Preview in Designer

If in the Windows Internet settings (**Connection Settings > LAN configuration**) a proxy is enabled, but "Bypass proxy settings for local addresses" is not checked, the image preview service, conversion service and live preview tab in the Designer will not work and exhibit the following issues:

- Images will be shows as 0 size boxes (no red 'X' is displayed)
- Live preview does not progress, and when re-activated reports "browsers is busy"

To fix the issue you must check the "Bypass proxy settings for local addresses" option.

### Merge\Weaver Engines when Printing

The print operation in the Designer will automatically detect whether the Merge\Weaver engines are available and display a message for the user to retry or cancel if not. Once the Merge\Weaver engine becomes available and the user presses retry the print operation will proceed as normal. This message can also occur in the following circumstances:

- If the server is offline and you are not using Proof Print
- On some occasions before the Print Wizard opens

### REST Calls for Remote Services

The Server will now accept REST calls for all remote services and will make commands wait indefinitely until the required engines become available. The Server will log when it is waiting for an engine and when it becomes available. Note that there is no way to cancel any commands other than stopping the Server.

### Print Content and Email Content in PlanetPress Workflow

In PlanetPress Workflow's Print Content and Email Content tasks, the option to Update Records from Metadata will only work for fields whose data type is set to String in the data model. Fields of other types will not be updated in the database and no error will be raised. This will be fixed in a later release.

## VIPP Output

Some templates set up with landscape orientation are being produced as portrait in VIPP. It can also sometimes be the case that text and images can be slightly displaced. These are known issues and will be addressed in a later release of Connect.

## Print Limitations when the Output Server is located on a different machine

The following limitation may occur when using the Print options from a Designer located on a different machine to the Output Server:

- The file path for the prompt and directory output modes is evaluated on both the client AND server side. When printing to a network share it must be available to BOTH the Designer and Server for the job to terminate successfully.
- The Windows printer must be installed on both the Server and Designer machines.
- When printing via the Server from a remote Designer, the output file remains on the Server machine. This is remedied by selecting “Output Local” in the Output Creation configuration.

## Overview

This document provides an overview of the new features and enhancements in PlanetPress Connect 1.4.n and PlanetPress Workflow 8.4.

### **Installing PlanetPress Connect 1.4.n and PlanetPress Workflow 8.4**

- PlanetPress Connect is released as a 64 Bit version only (with the exception of the Workflow, Fax, Search and Imaging modules).
- Full details on installing and licensing PlanetPress Connect and PlanetPress Workflow can be found in the online help in the installer.
- Note that both PlanetPress and PlanetPress Connect Workflow come with 30 day trial licenses by default.

### **Updating from PlanetPress Connect 1.1**

In order to upgrade from Connect Version 1.1 to Version 1.4.n via the Update Manager it is necessary to install a new version of the Objectif Lune Update Client. The next time you run your current Update Client it will show that there is an update available for itself. Simply click on

the download button in the dialog to install the new version of the Update Client. Note that it is no problem to run the update while the Client is open. It will automatically update itself.

Once you have done this, PlanetPress Connect 1.4.n will become available for download.

From Connect Version 1.2.0 onwards, the newer version of the Update Client was included with the Connect installation.

### **Print Only Version**

A Print Only license is available with version 1.4.n of PlanetPress Connect which allows legacy PlanetPress Suite 7 customers on OL Care to upgrade to Connect for a minimal fee. The license allows regular printing via the Print Wizard but runs Email and Web output in demo mode. For more information, please contact your local OL Customer Care or Sales team.

### **Templates Used in Workflow**

For improved performance we recommend resaving Workflow templates set up in the previous versions to run with PlanetPress Connect 1.4.n\Workflow 8.4.

### **Reduced Memory Version**

#### **Note**

This is **not** recommended for production.

It is now possible to install PlanetPress Connect on a machine with a minimum of 2 GB of RAM. The PlanetPress Connect Designer will automatically detect whether it has been installed on a machine with less than 4 GB of RAM and default to only using one internal Weaver and one internal merge engine on that system. The Server will also run using internal engines.

## Connect 1.4.2 Enhancements and Fixes

### Designer

- A blank page is no longer added to beginning of templates that use scripting to add pages from PDF files. This problem only appeared when saving to a new file from within Preview mode, or when generating output from Preview mode. (SHARED-44564)
- Image elements (<img>) referencing a PDF image would multiply when switching back and forth between Live and Preview modes, in both email and web contexts. This has now been fixed. (SHARED-44066)

### Email and Web Output

- Elements whose style was set to `display: none` would be removed from HTML output, rather than just not being displayed. This error has been fixed. (SHARED-44151)

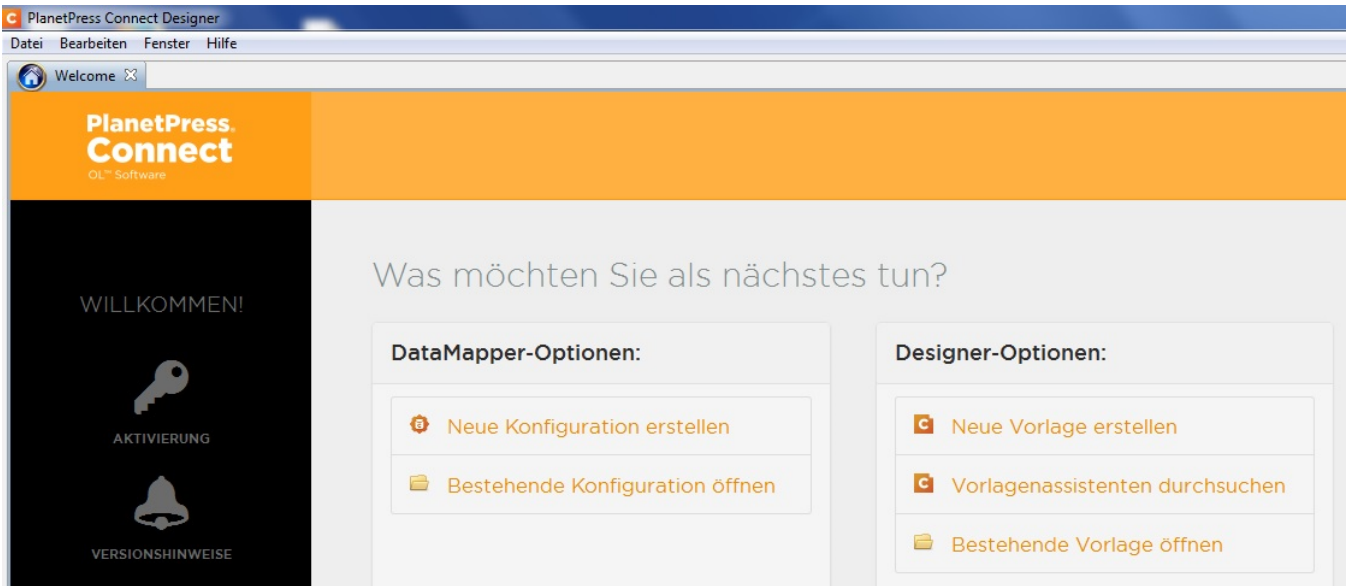
#### Note

Elements hidden via the Conditional Script wizard are removed from the output.

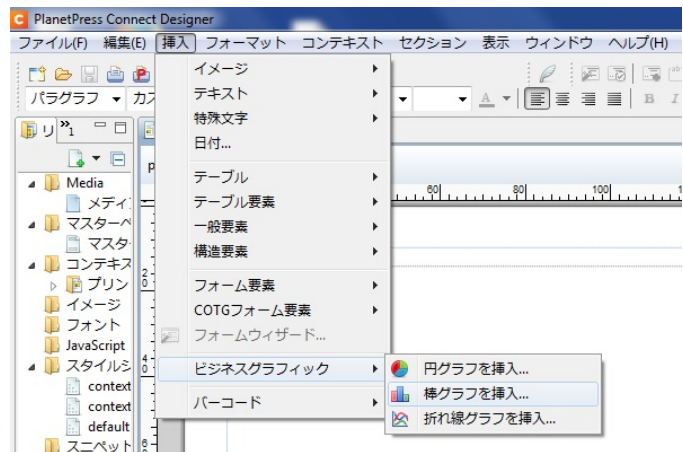
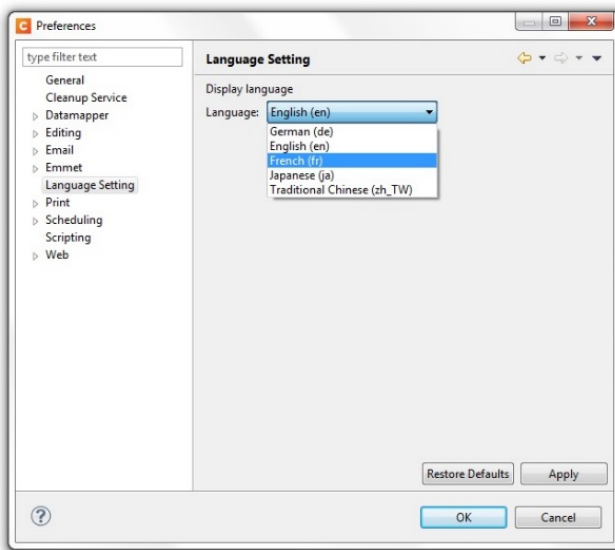
## Connect 1.4.1 New Features and Enhancements

### New Languages Added

The Connect user interface is now supported in Spanish, Italian, Portuguese and Chinese (Simplified) as well as English, French, German, Japanese and Chinese (Traditional). The default language remains English. Further languages will be introduced in later releases.



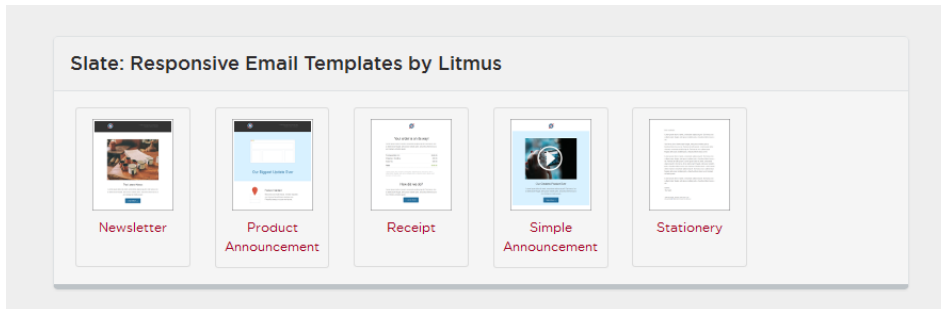
The language can be selected during the installation of Connect or via the Language Setting options in the Preferences dialog (note that Connect needs to be restarted in order to apply the selected language).



At present only the Connect user interface has been translated. Error messages and warnings will be translated for a later release.

## Welcome Screen Extended

- The Printer Definition Configs and HCF files available on the OL Connect website are now grouped by manufacturer, to simplify selection.
- Connect 1.4.1 also introduces **Responsive Email Templates**.



## Virtualisation

- Connect is now supported on the Microsoft Hyper-V and Hyper-V/Azure environments as well as the VMWare Workstation, Server, Player and ESX infrastructure environments.

## Modifying Connect Installations

- Connect 1.4.1 introduces the ability to Modify Connect installations, once Connect 1.4.1 has been installed.

## Connect 1.4.1 Designer Enhancements and Fixes

### Capture OnTheGo

- **Remote CSS** and **Javascript** resources can now be pre-fetched and shared between documents to avoid having to embed them in every template. (SHARED-39095)
- **Signature widgets** can now be marked as required. (SHARED-39833)
- **Annotations** can now be added to static images (not the picture widget). (SHARED-40369)

## Email Context

- Email context sections can be **enabled or disabled based upon data value**. (SHARED-33656)
- Email **port number** can now be specified as part of the host name. (SHARED-38008)
- **New template wizard** for Slate templates by Litmus. (SHARED-36843)

## Print Context

- Ability added to **mirror margins on back pages of Duplex jobs**, via Facing Pages selection added in Sheet Configuration dialog. (SHARED-40505)
- Can now suppress Master Page on duplex back pages, if there are no contents.

## Scripting

- Result can now be written as a JSON string into an attribute or text (instead of the field value). Useful for web contexts where a front-end script can easily read the value.
- User-defined formatting masks can now be used when outputting dates and numerical values.
- Conditional script set to "Show if Condition is true" will hide the object by default.
- Page range can be specified via scripting when setting PDF as background. (SHARED-36998)
- Matched elements are highlighted when hovering over scripts. (SHARED-38293)

## General

- Numerous improvements to Designer GUI and user-experience. These include:
  - A new "**Styles**" pane that displays the styles applied to the selected object.
  - **Image Selection** Dialog now inserts from resources, files or URL.
  - **Line number** option now available from any source view.
  - View menu now has entries to **switch between views** within the Workspace.
- Set any **PDF as background** for a Section. (SHARED-39880)
- Specify a **background image** and control its size, position and repeat mode. (SHARED-14522)

- Option to **generate JSON string from data model fields** to pass data record information to client side script. (SHARED-39337)
- **CSS Class name completion** suggests CSS classes based upon the current section. (SHARED-36870)
- **CSS Style inspector** allows full control over styles. (SHARED-22929)

## Connect 1.4.1 DataMapping Enhancements and Fixes

- **All fields can now be renamed** through the Data Model view. (SHARED-40116)
- SQL Server data mapping can now use **Windows Authentication** credentials. (SHARED-38999)
- Support added for **MS SQL Server** as the backend database. (SHARED-39959)

## Connect 1.4.1 Output Enhancements and Fixes

### Print Output

- Images, barcodes, OMRs and text can now all be added to the page at time of output generation.
- Ability to **add Metadata** to any output type (previously only PDF and AFP), for use within output Presets.
- **Static strings** can now be added to Metadata in job Presets.
- The **Output Path** in the Output Preset can now be set dynamically.
- New Color setting for the **Add Text** option. (SHARED-40830)
- **Monochrome** and **Dithering** support added to PCL output. (SHARED-39937)
- **Dithering** and **B&W threshold** values now supported in the Print Wizard. (SHARED-39520)
- New option to control **Compact Font Format** settings within PostScript. (SHARED-39902)

### Email Output

- **SMTP port** can now be customized when sending email.



## Connect 8.4.1 Workflow Enhancements and Fixes

- Major performance improvements when updating data records. (SHARED-38897)
- Stand-alone **Update Data Record** task allows data records to be updated in the database without having to create content. (SHARED-38867)
- **Update Records** operations performed in batches, allowing for unlimited number of records. (SHARED-38948)
- **All-In-One task** can now return the output file to the Workflow, to allow customising output destination(s). (SHARED-39434)
- **Job information** (such as Owner and Job Name) now sent to the printer in PostScript and Windows Driver output.

## Known Issues

### Installation Paths with Multi-Byte Characters

When installing the Chinese (Traditional or Simplified) or Japanese versions of Connect, if the user specifies an alternative installation path containing multi-byte/wide-char characters it can break some of the links to the Connect-related shortcuts in the Start Menu and cause an error to appear at the end of the installer. The workaround for the moment is to use the default installation path. The problem will be addressed in a later release.

### Switching Languages

Changing the language using the **Window>Preferences>Language Setting** menu option does not currently change all of the strings in the application to the selected language. This is a known issue and will be fixed in a later release.

In the meantime we offer the following workaround for anyone who needs to change the language:

1. Go to the .ini files for the Designer and Server Config:
  - C:\Program Files\Objectif Lune\OL Connect\Connect Designer\Designer.ini
  - C:\Program Files\Objectif Lune\OL Connect\Connect Server Configuration\ServerConfig.ini
2. Change the language parameter to the required one under Duser.language=en | fr | de | ja | zh

Only one of the above language tags should be selected. Once saved, Connect will appear in the selected language at next start-up.

### **GoDaddy Certificates**

When installing Connect offline, dialogs allow installing the GoDaddy certificates. Most users should use the default settings and click **Next**. In some cases, however, this may not work correctly. For this reason those users should activate **Place all certificates in the following store** and then select the **Trusted Root Certification Authorities** as the target certificate store.

### **MySQL Compatibility**

After installing Connect 1.4.n a downgrade to a Connect version older than Connect 1.3 or to a MySQL version earlier than 5.6.25 is not seamlessly possible. This is because the database model used in Connect 1.3 and 1.4.n (MySQL 5.6) is different to earlier versions. If you need to switch to an older version of Connect / MySQL, it is necessary to remove the Connect MySQL Database folder from "%ProgramData%\Connect\MySQL\data" before installing the older version.

### **PostScript Print Presets**


The print presets for PostScript were changed from Version 1.1 onwards meaning that some presets created in Version 1.0 or 1.0.1 may no longer work.

Any PostScript print preset from Version 1.0 that contains the following will not work in Version 1.4: \*.all[0].\*

Any preset containing this code will need to be recreated in Version 1.4.

### **Available Printer Models**

Note that only the single Printer Model (Generic PDF) will appear on the **Advanced** page of the **Print Wizard** by default.

To add additional printer models click on the settings  button next to the Model selection entry box.

Note that the descriptions of some of the printers were updated in version 1.2 meaning that if you had version 1.n installed, you may find that the same printer style appears twice in the list, but with slightly different descriptions.

For example the following printer types are actually identical:

- Generic PS LEVEL2 (DSC compliant)
- Generic PS LEVEL2 (DSC)

## External Resources in Connect

There are certain limitations on how external resources can be used in Connect. For example if you want to link a file (e.g., CSS, image, JavaScript etc.) from a location on the network but you do not want to have a copy of the file saved with the template you need to do the following:

1. The resource must be located where it can be accessed by all Servers/Slaves run as users. Failure to do this will cause the image to appear as a Red X in the output for all documents which were merged by engines which could not access the file. The job will terminate normally and the error will be logged.
2. The file must be referenced via a UNC path e.g.,  
file:///w2k8r2envan/z%20images/Picture/Supported/JPG/AB004763.jpg
  - UNC paths are required because the services will be unable to access mapped network drives (Windows security feature).
  - The engine processing the job will look on the local file system for the direct file path leading to the “resource not found” issue mentioned above.

### Warning

Important Note: The Designer itself and Proof Print do not use processes that run as services and they may find local files with non-UNC paths which can lead to the false impression that the resources are correct.

## Using Capture After Installing Workflow 8

If PlanetPress Connect Workflow 8 is installed alongside PlanetPress Suite Workflow 7, Capture can no longer be used within Workflow 7. The plugins are now registered uniquely to Workflow 8 and the messenger for Workflow 7 is taken offline. It is only possible to use Capture from PlanetPress Connect Workflow 8 thereafter.

## Capturing Spool Files After Installing Workflow 8

If PlanetPress Connect Workflow 8 is installed alongside PlanetPress Suite Workflow 7, the PlanetPress Suite 7 option to capture spool files from printer queues will no longer function. The solution is to use PlanetPress Connect Workflow 8 to capture spool files from printer queues.

## Colour Model in Stylesheets

The colour model of colours defined in a stylesheet can sometimes change after editing the stylesheet. This is a known issue and will be addressed in a subsequent release.

## Online Help Links Point to Introductory Page

Context sensitivity for the online help is not yet enabled in Connect. All links and F1 calls point to the introductory page, where you can Search on keywords to bring up Help pages relating to the topic.

Context sensitivity will be introduced in a subsequent release of Connect.

## Image Preview in Designer

If in the Windows Internet settings (**Connection Settings > LAN configuration**) a proxy is enabled, but "Bypass proxy settings for local addresses" is not checked, the image preview service, conversion service and live preview tab in the Designer will not work and exhibit the following issues:

- Images will be shown as 0 size boxes (no red 'X' is displayed)
- Live preview does not progress, and when re-activated reports "browsers is busy"

To fix the issue you must check the "Bypass proxy settings for local addresses" option.

## MergeWeaver Engines when Printing

The print operation in the Designer will automatically detect whether the MergeWeaver engines are available and display a message for the user to retry or cancel if not. Once the MergeWeaver engine becomes available and the user presses retry the print operation will proceed as normal. This message can also occur in the following circumstances:

- If the server is offline and you are not using Proof Print
- On some occasions before the Print Wizard opens

## **REST Calls for Remote Services**

The Server will now accept REST calls for all remote services and will make commands wait indefinitely until the required engines become available. The Server will log when it is waiting for an engine and when it becomes available. Note that there is no way to cancel any commands other than stopping the Server.

## **Print Content and Email Content in PlanetPressWorkflow**

In PlanetPress Workflow's Print Content and Email Content tasks, the option to Update Records from Metadata will only work for fields whose data type is set to String in the data model. Fields of other types will not be updated in the database and no error will be raised. This will be fixed in a later release.

## **VIPP Output**

Some templates set up with landscape orientation are being produced as portrait in VIPP. It can also sometimes be the case that text and images can be slightly displaced. These are known issues and will be addressed in a later release of Connect.

## **Print Limitations when the Output Server is located on a different machine**

The following limitation may occur when using the Print options from a Designer located on a different machine to the Output Server:

- The file path for the prompt and directory output modes is evaluated on both the client AND server side. When printing to a network share it must be available to BOTH the Designer and Server for the job to terminate successfully.
- The Windows printer must be installed on both the Server and Designer machines.
- When printing via the Server from a remote Designer, the output file remains on the Server machine. This is remedied by selecting "Output Local" in the Output Creation configuration.

# Legal Notices and Acknowledgements

PlanetPress Connect, Copyright © 2017, Objectif Lune Inc.. All rights reserved.

The license agreements for the associated open source third party components can be downloaded [here](#).

This application uses the following third party components:

- **Adobe PDF Library** which is either a registered trademark or trademark of Adobe Systems Incorporated in the United States and/or other countries.
- **Adobe XMP Core** Copyright © 1999 - 2010, Adobe Systems Incorporated. All rights reserved.
- **Eclipse Persistence Services Project (EclipseLink)**, Copyright © 2007, Eclipse Foundation, Inc. and its licensors. All rights reserved. This is distributed under the terms of the Eclipse Public License Version 1.0 and Eclipse Distribution License Version 1.0.
- **Fugue Icons** by [Yusuke Kamiyamane](#) which are distributed under the terms of the [Creative Commons Attribution 3.0 License](#).
- **Gecko** which is distributed under the terms of the Mozilla Public License Version 2.0. Information on obtaining Gecko can be found on the following page: [https://wiki.mozilla.org/Gecko:Getting\\_Started](https://wiki.mozilla.org/Gecko:Getting_Started)
- **Glassfish Java Mail** which is licensed under the terms of the Common Development and Distribution License (CDDL) Version 1.0. Information on how to download the Glassfish source can be obtained from here: <https://wikis.oracle.com/display/GlassFish/Java+EE+7+Maven+Coordinates>
- **Hamcrest Matchers** Copyright © 2000-2006, www.hamcrest.org. All rights reserved.
- **HyperSQL**, Copyright © 2001-2010, The HSQL Development Group. All rights reserved.
- **ICU4J 4.4.2** Copyright © 1995-2013 International Business Machines Corporation and others. All rights reserved.
- **J2V8** which is distributed under the terms of the Eclipse Public License Version 1.0. The source code for J2V8 can be obtained from the following location: <https://github.com/eclipsesource/j2v8>

- **Jacob Java Com Bridge** which is licensed under the terms of the GNU Lesser General Public License Version 2. The source code for this can be obtained from the following location: <http://sourceforge.net/projects/jacob-project/files/jacob-project/>
- **JavaCraft JSch** Copyright © 2002 - 2012 Atsuhiko Yamanaka, JCraft Inc. All rights reserved.
- **JavaSysMon** Copyright © 2009 ThoughtWorks, Inc. All rights reserved.
- **JavaX Mail** which is distributed under the terms of the Common Development and Distribution License (CDDL) Version 1.1. The source code for this can be obtained from the following location: <https://java.net/projects/javamail/downloads/directory/source>
- **Java XmlHttpRequest** which is licensed under the terms of the GNU Lesser General Public License Version 2.1. The source code for this can be obtained from the following location: <https://github.com/objectifluneCA/java-XmlHttpRequest>
- **Jersey** which is distributed under the terms of the Common Development and Distribution License (CDDL) Version 1.1. Information on how to obtain the source code can be found at the following location: <http://repo1.maven.org/maven2/org/glassfish/jersey/jersey-bom>
- **jersey-json-1.13** which is licensed under the terms of the Common Development and Distribution License (CDDL) Version 1.1. Information on how to obtain the source code can be found at the following location: <http://mvnrepository.com/artifact/com.sun.jersey/jersey-json/1.13-b01>
- **Jersey Multipart** which is distributed under the terms of the Common Development and Distribution License (CDDL) Version 1.1. Information on how to obtain the source code can be found at the following location: <http://repo1.maven.org/maven2/org/glassfish/jersey/jersey-bom>
- **JGoodies Forms, JGoodies Binding and JGoodies Looks**, Copyright © 2002-2013 JGoodies Software GmbH. All rights reserved.
- **JNA Version 3.5.1** which is distributed under the terms of the GNU Lesser General Public License Version 2.1. The source code for this can be obtained from the following location: <https://github.com/twall/jna/releases>
- **Junit** which is distributed under the terms of the Eclipse Public License Version 1.0. The source code for Junit can be obtained from the following location: <https://github.com/junit-team/junit/tree/master/src>
- **Mimepull** which is distributed under the terms of the Common Development and Distribution License (CDDL) Version 1.1. The source code for this can be obtained from the following location: <https://maven.java.net/content/repositories/releases/org/jvnet/mimepull/mimepull/>

- **Objectweb ASM**, Copyright © 2000-2011 INRIA, France Telecom. All rights reserved.
- **Relique CSV Driver** which is licensed under the terms of the Lesser General Public License Version 2.1. The source code can be obtained from the following location: <https://sourceforge.net/p/csvjdbc/code/ci/csvjdbc-1.0.31/tree/>
- **Rhino 1.7R4 and 1.7.7.1** which are licensed under the terms of the Mozilla License Version 2.0. The source code for these can be obtained from the following location: [https://developer.mozilla.org/en-US/docs/Mozilla/Projects/Rhino/Download\\_Rhino](https://developer.mozilla.org/en-US/docs/Mozilla/Projects/Rhino/Download_Rhino)
- **Saxon** which is distributed under the terms of the Mozilla Public License Version 2.0. The source code for this can be obtained from the following location: <http://sourceforge.net/projects/saxon/files/Saxon-HE/9.6/>
- **Servlet API** developed by Sun as part of the Glassfish project and licensed under the terms of the Common Development and Distribution License (CDDL) Version 1.0. Information on how to download the Glassfish source (as part of Java EE platform) can be obtained from here: <https://wikis.oracle.com/display/GlassFish/Java+EE+7+Maven+Coordinates>
- **Simple Logging Facade for Java (SLF4J)** Copyright © 2004-2017 QOS.ch. All rights reserved.
- **Spring Framework** which is distributed under the terms of the Apache Software License Version 2.0. This product includes subcomponents with separate copyright notices and license terms.
- **Springsource JavaX Mail** which is distributed under the terms of the Common Development and Distribution License (CDDL) Version 1.0. The source code for this can be obtained from the following location: <http://ebr.springsource.com/repository/app/bundle/version/detail>
- **Web Services Description Language for Java** which is distributed under the terms of the Common Public License v 1.0. The source code for this can be obtained from the following location: <http://wsdl4j.cvs.sourceforge.net/viewvc/wsdl4j/>
- **XULRunner** which is distributed under the terms of the Mozilla Public License Version 2.0. The source code for this can be obtained from the following location: <http://ftp.mozilla.org/pub/mozilla.org/xulrunner/releases/latest/source/>
- **zziplib** which is licensed under the terms of the Mozilla License Version 1.1. The source code for this can be obtained from the following location: <http://sourceforge.net/projects/zziplib/files/zziplib13/>



- **7-Zip SFX** which is licensed under the terms of the GNU Lesser General Public License Version 2.1. The source code for this can be obtained from the following location:  
<https://github.com/chrislake/7zsfxmm>

Portions of certain libraries included in this application which are distributed under the terms of the Mozilla Public License have been modified. To obtain copies of the modified libraries please contact your local Objective Lune Support team.

This application also uses the following components which are distributed under the terms of the **Apache Software License Version 2.0**:

- Apache Ant
- Apache Axis
- Apache CFX
- Apache Commons Beanutils
- Apache Commons CLI
- Apache Commons Codec
- Apache Commons Collections
- Apache Commons Configuration
- Apache Commons DBCP
- Apache Commons Digester
- Apache Commons Discovery
- Apache Commons FileUpload
- Apache Commons Imaging
- Apache Commons IO
- Apache Commons Lang
- Apache Commons Logging
- Apache Commons Net
- Apache Commons Pool
- Apache Commons Text
- Apache Commons Validator
- Apache Commons VFS
- Apache Derby
- Apache Felix and dependencies

- Apache Geronimo
- Apache Jakarta HttpClient
- Apache Log4j
- Apache Neethi
- Apache OpenCMIS
- Apache POI
- Apache ServiceMix
- Apache Tomcat
- Apache WSS4J
- Apache Xalan
- Apache Xerces2 Java Parser
- Apache XMLGraphics
- Apache XML-RPC
- Barcode4j
- Google Collections
- Google GSON
- Hibernate Validator
- Jetty
- LMAX Disruptor
- Objenesis
- OpenCSV
- OPS4J Pax Web
- org.json.simple
- Quartz Scheduler
- Spring Dynamic Modules
- StAX
- UCanAccess
- XMLBeans

### **Eclipse Technology:**

This Software includes unmodified Eclipse redistributables, which are available at [www.eclipse.org](http://www.eclipse.org). The Eclipse redistributables are distributed under the terms of the Eclipse Public License - v 1.0 that can be found at <https://www.eclipse.org/legal/epl-v10.html>.

### **VSS Java FreeMarker:**

This product includes software developed by the Visigoth Software Society (<http://www.visigoths.org/>).

This includes the following subcomponents that are licensed by the Apache Software Foundation under the Apache License, Version 2.0:

- freemarker/ext/jsp/web-app\_2\_2.dtd
- freemarker/ext/jsp/web-app\_2\_3.dtd
- freemarker/ext/jsp/web-app\_2\_4.xsd
- freemarker/ext/jsp/web-app\_2\_5.xsd
- freemarker/ext/jsp/web-jsptaglibrary\_1\_1.dtd
- freemarker/ext/jsp/web-jsptaglibrary\_1\_2.dtd
- freemarker/ext/jsp/web-jsptaglibrary\_2\_0.xsd
- freemarker/ext/jsp/web-jsptaglibrary\_2\_1.xsd

### **Java SE framework and platform:**

This application uses the Java SE framework and platform which is distributed under the terms of the Oracle Binary Code License Agreement for the Java SE Platform Products and Java FX. Copyright 2013, Oracle America, Inc. All rights reserved.

Use is subject to license terms. ORACLE and JAVA trademarks and all ORACLE- and JAVA-related trademarks, service marks, logos and other brand designations are trademarks or registered trademarks of Oracle in the U.S. and other countries.

*Use of the Commercial Features for any commercial or production purpose requires a separate license from Oracle. "Commercial Features" means those features identified Table 1-1 (Commercial Features In Java SE Product Editions) of the Java SE documentation accessible at <http://www.oracle.com/technetwork/java/javase/documentation/index.html>.*

### **Further Components:**

- This product includes software developed by the **JDOM Project** (<http://www.jdom.org/>).
- Portions of this software are copyright © 2010 **The FreeType Project** ([www.freetype.org](http://www.freetype.org)). All rights reserved.
- This product includes software developed by **JSON.org** (<http://www.json.org/java/index.html>).

## Copyright Information

Copyright © 1994-2018 Objectif Lune Inc. All Rights Reserved.

No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any other language or computer language in whole or in part, in any form or by any means, whether it be electronic, mechanical, magnetic, optical, manual or otherwise, without prior written consent of Objectif Lune Inc.

Objectif Lune Inc. disclaims all warranties as to this software, whether expressed or implied, including without limitation any implied warranties of merchantability, fitness for a particular purpose, functionality, data integrity or protection.

PlanetPress, PReS and PrintShop Mail are registered trademarks of Objectif Lune Inc.

[Click to download the EULA as PDF](#)