Automation Services 8.1 Workflow Reference

Contents

Legal notices

Introduction

This Guide explains the functionality of the workflows included with Automation Services for Quark Publishing System® (QPS®) 8.1.

Where we're coming from

This book assumes you are familiar with your computer and know how to:

- Launch an application
- Open, save, and close files
- Use menus, dialog boxes, and palettes
- Use the mouse, keyboard commands, and modifier keys

If you need help performing any of these tasks, consult the documentation resources (user or reference guides) provided with your computer.

Conventions in this book

Formatting conventions in this guide highlight information to help you quickly find what you need.

- Bold type style: The names of all dialog boxes, fields, and other controls are set in bold type. For example: "Click **Storage** in the **Administration** pane."
- References: In descriptions of features, parenthetical references guide you in accessing those features. For example: "The **System Storage** controls (**Administration** pane) let you designate asset storage."
- Arrows: You will often see arrows (>), which map the path to a feature. For example: "Choose **Administration** > **User Profiles** to add a user."
- Icons: Although many tools and buttons are referenced by name, which you can see by displaying ToolTips, in some cases icons are shown for easy identification.

- Cross-platform issues: Some labels, buttons, key combinations, and other aspects of QPS client applications differ between Mac OS® and Windows® because of user interface conventions or other factors. In such cases, both the Mac OS and Windows versions are presented, separated by a slash, with the Mac OS version presented first. For example, if the Mac OS version of a button is labeled **Select**, and the Windows version is labeled **Browse**, you are directed to "Click **Select/Browse**." More complex cross-platform differences are mentioned in notes or parenthetical statements.

➡ Notes provide helpful information about particular features and general techniques for using the software.

Understanding workflows

A workflow is a predefined sequence of actions for creating output. You can also think of a workflow as a framework on which you can build an automated task. For more information, see "Understanding workflows" in *A Guide to Automation Services*.

Workflows

The following topics describe the workflows currently available from Quark for Automation Services.

➡ In order to work correctly, Automation Services needs to have sufficient rights to read and write to the folders it works with. For more information, see "Setting access privileges and idle time-out" in *A Guide to Automation Services*.

XML to QuarkCopyDesk

The XML to QuarkCopyDesk® workflow lets you transform content from QuarkXPress® projects, QuarkCopyDesk articles, and XML sources into QuarkCopyDesk articles without using an article template.

If the output of the transformation for this workflow includes a QPSEntityModel mapping component, Automation Services uses the default Job Jackets file (if any) for the target main category/subcategory. Otherwise, Automation Services uses the default server-level Job Jackets file for QPS.

- **Transformation**: Choose a transformation that has QuarkCopyDesk as an output format, such as **DITA to QuarkCopyDesk Article with Template**.
- **Image Folder Paths**: Specify the path(s) where any images required by the workflow can be found. When the automation profile executes, the software searches these paths for any images referenced in the transformation's XML output, attaches them to the article, and checks them into QPS (if necessary). Note that if you specify a path in the file system, you must use a valid UNC path to which the Automation Services server has read access. If Automation Services cannot locate a picture file, it substitutes a default picture with the same name as the missing picture. The default picture can be found at `[application folder]\AutomationServices_Data\Resources\ProfileManagement\FallbackImage.jpg`.
- **Save Location**: Specify where you want the final file to be put. You can save the file to the file system or check it into QPS. If you check the file into QPS, the transformation must supply the corresponding metadata in a **QPSEntityModel** output. Note that if you specify a path in the file system, you must use a valid UNC path to which the Automation Services server has write access, and the path must include the name of the output article. Pictures are stored in a "Pictures" folder at the same level as the output article.

XML to QuarkCopyDesk From Template

The XML to QuarkCopyDesk From Template workflow lets you transform content from QuarkXPress projects, QuarkCopyDesk articles, and XML sources into QuarkCopyDesk articles with an article template.

- **Template**: Choose a QuarkCopyDesk article template.
- **Transformation**: Choose a transformation that accepts a QuarkCopyDesk template and has QuarkCopyDesk as an output format, such as **NewsML to QuarkCopyDesk Article with Template**.
- **Image Folder Paths**: Specify the path(s) where any images required by the workflow can be found. When the automation profile executes, the software searches these paths for any images referenced in the transformation's XML output, attaches them to the article, and checks them into QPS (if necessary). Note that if you specify a path in the file system, you must use a valid UNC path to which the Automation Services server has read access. If Automation Services cannot locate a picture file, it substitutes a default picture with the same name as the missing picture. The default picture can be found at `[application folder]\AutomationServices_Data\Resources\ProfileManagement\FallbackImage.jpg`.
- **Save Location**: Specify where you want the final file to be put. If you check the file into QPS, the transformation must supply the corresponding metadata in a **QPSEntityModel** output. Note that if you specify a path in the file system, you must use a valid UNC path to which the Automation Services server has write access, and the path must include the name of the output article. Pictures are stored in a "Pictures" folder at the same level as the output article.

XML to QuarkXPress

The XML to QuarkXPress workflow lets you transform content from QuarkXPress projects, QuarkCopyDesk articles, and XML sources into QuarkXPress layouts with a QuarkXPress template.

- **Template**: Choose a QuarkXPress article template.
- **Transformation**: Choose a transformation that accepts a QuarkXPress template and has QuarkXPress as an output format, such as **DocBook to QuarkXPress Template to PDF**.
- **Image Folder Paths**: Specify the path(s) where any images required by the workflow can be found. When the automation profile executes, the software searches these paths for any images referenced in the transformation's XML output, attaches them to the layout, and checks them into QPS (if necessary). Note that if you specify a path in the file system, you must use a valid UNC path to which the Automation Services server has read access. If Automation Services cannot locate a picture file, it substitutes a default picture with the same name as the missing picture. The default picture can be found at `[deployment directory]\AutomationServices_Data\Resources\ProfileManagement\FallbackImage.jpg`.
- **Save Location**: Specify where you want the final file to be put. If you check the file into QPS, the transformation must supply the corresponding metadata in a **QPSEntityModel**

output. Note that if you specify a path in the file system, you must use a valid UNC path to which the Automation Services server has write access, and the path must include the name of the output project. Pictures are stored in a "Pictures" folder at the same level as the output project.

➡ If a transformation in an automation profile makes changes to an existing project and then checks it in as a new revision, you need to be aware of how Automation Services deals with attachments in such projects. If the automation profile makes changes to a box that has an attachment, the attachment is detached in the new revision, unless the Automation Profile includes a **QPSEntityModel** mapping component for the attached article. If the automation profile produces a new project rather than a new revision of an existing project, *all* attached articles are detached unless they have corresponding **QPSEntityModel** mapping components.

XML to XHTML

The XML to XHTML workflow lets you transform content from QuarkXPress projects, QuarkCopyDesk articles, and XML sources into XHTML format. You can then automatically upload the resulting XHTML content into Drupal.

- **Transformation**: Choose a transformation that accepts an XML input and has XHTML as an output format, such as **DITA Topic to XHTML to Drupal** or **QuarkCopyDesk to XHTML to Drupal**.
- **Image Folder Paths**: Specify the path(s) where any images required by the workflow can be found. Note that if you specify a path in the file system, you must use a valid UNC path to which the Automation Services server has sufficient access rights to read files.
- **Image Format**: Specify the format and (if applicable) quality level for pictures in the workflow.
- **Save Location**: Specify where you want the final files to be put. You can save the output to the file system or submit it as a "Page" to a running instance of the Drupal Web content management system.

If you save the output to the file system, you must use a valid UNC path to which the Automation Services server has write access, and the path must include the name of the output XHTML file. Pictures are stored in an "Images" folder at the same level as the output XHTML file.

If you submit the output to Drupal, you must specify the node location of the file that you are uploading. All nodes created by this workflow will be placed at that location, in a node with the same name as the trigger asset. For example, if you specify the location `news/sports` and the name of the trigger asset is "Tennis.xml," the content is checked in at `news/sports/Tennis`. The title of each page is determined by the `<title>` tag in the XHTML output.

➡ If an automation profile creates a new version of a page that already exists in Drupal, Drupal will create a new version of only the page itself; any pictures used in the page will simply be replaced.

➡ Drupal locations should not start with a slash (/).

If you have trouble submitting content to Drupal, a good place to start troubleshooting is the "AutomationServices.Server.config" file (located at `[application folder]\AutomationServices_Data\Resources\Configuration`). Locate the "scriptRelativeUri" `<entry>` element and verify that the "Quark Automation Services Content Uploader" page can be found in this directory (for more information, see "Installing Drupal for Automation Services" in the Automation Services ReadMe file). Locate the "imageFolderPath" `<entry>` element and make sure this path points to the directory where you want picture files to be uploaded.

Another setting to check is the `imagefolderpath` entry in the "AutomationServices.Server.config" file (located at `[application folder]\Server\AutomationServices_Data\Resources\Configuration`). This entry should be set to the same value as the `IMAGES_ROOT` value in the "Drupal Content Uploader.php" file.

XML to XML

The XML to XML workflow lets you transform content from QuarkXPress projects, QuarkCopyDesk articles, and XML sources into any valid XML output format.

- **Transformation**: Choose a transformation that accepts an XML input and has XML as an output format.
- **Save Location**: Specify where you want the final file to be put. If you check the file into QPS, the transformation must supply the corresponding metadata in a **QPSEntityModel** output. Note that if you specify a path in the file system, you must use a valid UNC path to which the Automation Services server has write access, and the path must include the name of the output XML file.

If you save the output to the file system, you must use a valid UNC path to which the Automation Services server has write access, and the path must include the name of the output XML file.

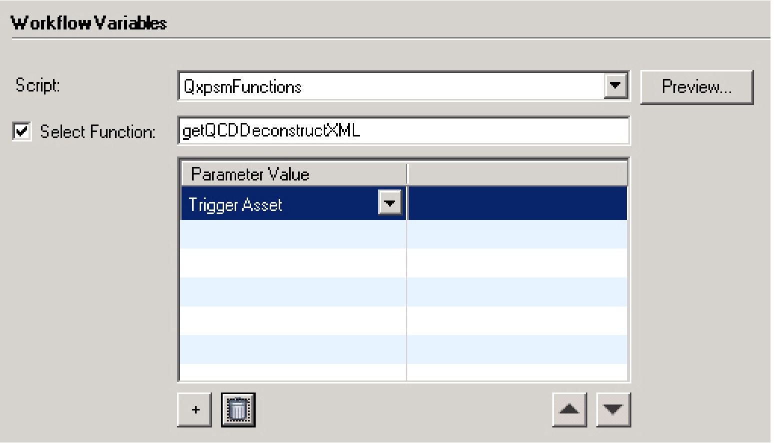
Export from QuarkXPress Project

The Export from QuarkXPress workflow lets you export layouts from QuarkXPress projects in various formats. You can export Print layouts in PDF and SWF format and Interactive layouts in SWF format.

- **Image Folder Paths**: Specify the path(s) where any images required by the workflow can be found. This parameter is used only if the QuarkXPress project originates from the file system. If the project is in QPS, its picture attachments (if any) are used. If the project is in the QuarkXPress Server document pool, Automation Services assumes that the pictures are also in the document pool. Note that if you specify a path in the file system, you must use a valid UNC path to which the Automation Services server has read access.
- **Rendition**: Choose the name of the output style you want to use for PDF rendering. If you do not choose a value, Automation Services uses the default QuarkXPress Server render type.
- **Save Location**: Specify where you want the final file to be put. Note that if you specify a path in the file system, you must use the valid UNC path of a directory which the Automation Services server has write access. Automation Services auto-generates output file names using the formula `[ProjectName]_[LayoutName].[FormatExtension]`.

Execute QPS Script

The Execute QPS Script workflow lets you execute either a single function in a QPS script or the entire script.



Workflow Variables

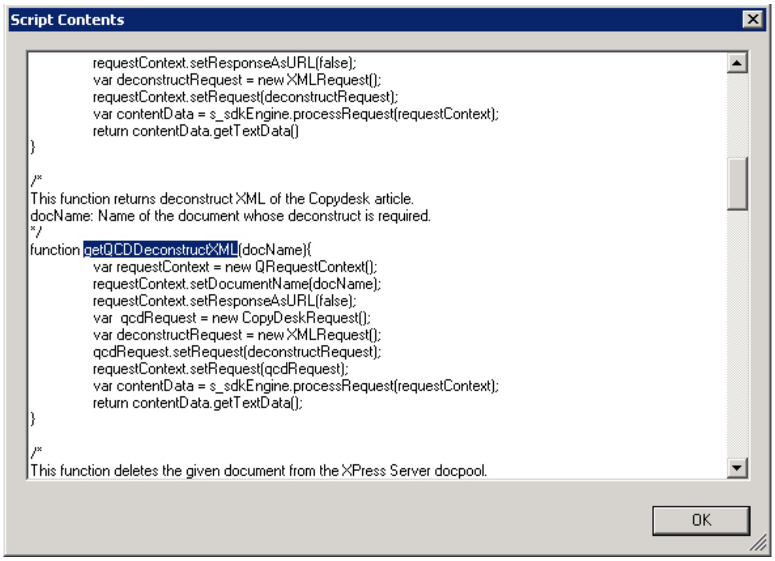
Script: QxpsmFunctions

☑ Select Function: getQCDDeconstructXML

Parameter Value	
Trigger Asset	

Workflow Variables area for Execute QPS Script workflow

- **Script**: Select the name of the target script.
- **Preview**: Displays the selected script in the **Script Contents** dialog box. If you only want to execute a single function, this can help you to locate the function you want and determine what its parameters are and what order they occur in.



```
            requestContext.setResponseAsURL(false);
            var deconstructRequest = new XMLRequest();
            requestContext.setRequest(deconstructRequest);
            var contentData = s_sdkEngine.processRequest(requestContext);
            return contentData.getTextData()
}

/*
This function returns deconstruct XML of the Copydesk article.
docName: Name of the document whose deconstruct is required.
*/
function getQCDDeconstructXML(docName){
            var requestContext = new QRequestContext();
            requestContext.setDocumentName(docName);
            requestContext.setResponseAsURL(false);
            var  qcdRequest = new CopyDeskRequest();
            var deconstructRequest = new XMLRequest();
            qcdRequest.setRequest(deconstructRequest);
            requestContext.setRequest(qcdRequest);
            var contentData = s_sdkEngine.processRequest(requestContext);
            return contentData.getTextData();
}

/*
This function deletes the given document from the XPress Server docpool.
```

Script Contents dialog box

- **Select Function**: If you want to execute a single function, check this box and enter the function name (without parameters) in the corresponding field.
- **Parameter Value** list: The rows in this list correspond to the parameters of the selected function. The first row corresponds to the functions first parameter, the second row to the function's second parameter, and so forth. To add a row, click +. To pass a the trigger asset to a parameter, choose **Trigger Asset** from the drop-down menu for that row. To pass a literal value to a parameter, choose **Other** from the drop-down menu for that row and then double-click in the corresponding field to the right and enter the value.

If you plan to use the trigger asset, a template, or some other file or asset as a parameter value for a script, you must make sure the script has been set up to handle that file or asset. To do so, load the "ScriptWorkflowUtil.js" script and use its `createFilePathContainer()` method. For example, if you wanted to write a simple script that writes the name of the trigger asset to the console, it might look like this:

```
load("AutomationServicesUtil");
...
function displayFileName(filePathParam)
{
  // Create a filePathContainer from the file path parameter
  var filePathContainer = createFilePathContainer(filePathParam);
  if (filePathContainer.assetId != null) {

    // Get the file name and write it to the console
    var assetName = getAssetName(filePathContainer.assetId);
    print(assetName);
  }
}
```

The createFilePathContainer function (supplied in the "AutomationServicesUtil.js" script) takes a file reference that has been passed from Automation Server and converts it to a filePathContainer object. You can then access the path to the asset with `filePathContainer.filePath` and get the ID of the asset with `filePathContainer.assetId`.

For more information, see the documentation in the "ScriptWorkflowUtil.js" script.

Index