

For Windows and Mac

FileMaker[®] Developer 6

Developer's Guide



© 1998, 2000-2002 FileMaker, Inc. All Rights Reserved.

FileMaker, Inc.

5201 Patrick Henry Drive

Santa Clara, California 95054

www.filemaker.com

FileMaker documentation is copyrighted. You are not authorized to make additional copies or distribute this documentation without written permission from FileMaker. You may use this documentation solely with a valid licensed copy of FileMaker software.

FileMaker is a trademark of FileMaker, Inc., registered in the U.S. and other countries, and ScriptMaker and the file folder logo are trademarks of FileMaker, Inc. Portions of some screen shots are reprinted by permission from Microsoft Corporation. Portions of some screen shots are copyright 1996- 2000 Netscape Communications Corp. All rights reserved. These screen shots may not be reprinted or copied without the express written permission of Netscape. All other trademarks are the property of their respective owners.

This software is based in part on the work of the Independent JPEG group. Portions of this software are ©1990 Access Softek and © 1991-2002 DataDirect Technologies. All rights reserved. All persons and companies listed in the examples are purely fictitious and any resemblance to existing persons and companies is purely coincidental. Mention of third party companies and products is for informational purposes only and does not constitute an endorsement. FileMaker assumes no responsibility with regard to the selection, performance, or use of these products. All understandings, agreements or warranties, if any, take place directly between the vendor and prospective users.

THIS PAGE INTENTIONALLY LEFT BLANK

Contents

Chapter 1

Welcome to FileMaker Developer

About the Developer edition of FileMaker Pro	1-1
About this guide	1-1
Registration and customer support	1-1
About the installation code	1-2
Hardware and software requirements	1-2
Networking requirements	1-3
Web publishing requirements	1-3
Requirements for advanced features (Windows)	1-3
Requirements for advanced features (Mac OS)	1-3
Features not supported In Mac OS X	1-4
Opening files from previous versions	1-4
Updating your User Dictionary	1-5
Read Me file	1-5
Electronic documentation	1-5
Abiding by the license agreement	1-6
About the TechInfo database	1-6

Chapter 2

Installing FileMaker Developer in Windows

Installation notes	2-1
Installing FileMaker Developer	2-1
Installing FileMaker Developer from the command line	2-2
Where files are stored on your hard disk	2-2
Uninstalling, modifying, or repairing FileMaker Developer	2-2
Adjusting network software	2-3
Changing the network protocol	2-3
Changing the IPX/SPX frame type	2-3
Changing the FileMaker Pro cache size	2-4
Where to go from here	2-4

Chapter 3

Installing FileMaker Developer in the Mac OS

Installation notes	3-1
Installing FileMaker Developer	3-1
Installing ODBC drivers in Mac OS X	3-2
Where files are stored on your hard disk	3-2
Accessing files from other user accounts (Mac OS X)	3-2
Choosing a network protocol (Mac OS 9.x or earlier)	3-3
Adjusting memory	3-3
Adjusting the FileMaker Pro cache (Mac OS X)	3-3
Adjusting memory requirements (Mac OS 9.x or earlier)	3-4
Enabling the FileMaker Pro Web Companion (Mac OS X)	3-5
Configuring the Web Companion for use with ports 1024 and higher	3-6
Accessing databases that are published to the Web	3-7
Where to go from here	3-7

Chapter 4

Creating a database solution

Overview of preparing your solution files	4-2
Considerations for a runtime database solution	4-2
Considerations for Kiosk mode	4-3
Displaying a database in Kiosk mode	4-3
Navigating in Kiosk mode	4-4
Closing a Kiosk mode solution	4-4
Example Kiosk solution	4-4
Using scripts to control your solution	4-5
Creating startup scripts	4-5
Centering a database window in the Kiosk screen	4-6
Emulating menu commands and window controls	4-6
Creating dynamic buttons	4-6

Protecting your database solution files	4-7	Table of single-line elements	5-6
Providing user documentation	4-8	Valid values for theme attributes	5-7
Creating an About layout	4-8	Specifying default values for themes	5-9
Creating a custom Help layout or file	4-9	Finding values for patterns and colors	5-9
Providing What's This? Help (Windows)	4-10	Adding comments to your document	5-10
Including printed documentation	4-10	Checking your theme document for errors	5-11
Design tips for cross-platform solutions	4-11	Chapter 6	
Creating a consistent appearance	4-11	Using the FileMaker Developer Tool	
Simulating outline and shadow text styles	4-11	About the solution examples	6-1
Using common character sets	4-12	Using the FileMaker Developer Tool	6-1
Designing text layouts for cross-platform solutions	4-12	Binding your databases into a runtime database solution	6-6
Using a common color palette	4-13	Procedure for binding files	6-6
Using graphics in cross-platform solutions	4-13	Specifying a runtime solution name and binding key	6-7
Using QuickTime movies in cross-platform solutions	4-13	Assigning a three-character filename extension	6-8
Showing the status bar in Windows	4-14	Binding files for cross-platform solutions	6-9
Using separate scripts for printing	4-14	Modifying bound runtime files	6-9
Using the Status (CurrentPlatform) function	4-14	Creating Kiosk-mode solutions	6-9
Creating platform-specific scripts	4-15	Renaming your databases	6-10
Your responsibilities as a developer	4-15	Procedure for renaming files	6-10
Testing before and after creating your solution	4-16	Choosing filenames for your runtime database solution	6-10
Converting and upgrading solution files	4-16	Removing design access to your databases	6-11
Importing data into an upgraded runtime database solution	4-17	Steps for preventing database modification	6-11
Chapter 5		Customizing the About, Help, and Scripts menus	6-11
Creating custom layout themes		Adding a custom script to an About menu command	6-11
Modifying a FileMaker Pro theme	5-1	Adding a custom Help script command to the Help menu	6-13
Basic requirements for a theme file	5-3	Renaming the Scripts menu	6-14
Minimum required XML elements for themes	5-3	Adding the FileMaker Pro extension to database filenames	6-14
Removing elements from a theme file	5-4	Saving your settings in the Developer Tool	6-14
XML elements for layout parts	5-4	Using the Solution Options text file	6-15
XML elements for text	5-5	Saving a set of options	6-15
Description of XML elements and their theme attributes	5-5	Modifying the Solution Options text file	6-15
Table of multi-line elements	5-5		

Chapter 7

Distributing FileMaker Pro runtime database solutions

Organizing your runtime database solution components	7-1
Generated DLL files for Windows runtime database solutions	7-2
System files for Windows solutions	7-2
Solution Extras files for Mac OS solutions	7-3
FileMaker Extensions files for Mac OS solutions	7-3
Choosing the distribution method	7-4
Requirements for distributing on floppy disks	7-4
Using a custom installation program	7-4
Using a compression utility program	7-4
What your users need	7-4
Sharing your solution over a network	7-5
Documenting the installation procedures	7-5
Starting your runtime database solution	7-5
Recovering damaged files	7-6
Distributing updates to your runtime database solution	7-6

Chapter 8

Publishing your database on the Web

Types of web publishing	8-1
Custom web publishing with XML	8-1
Custom web publishing with JDBC	8-1
Custom web publishing with CDML	8-2
Instant Web Publishing	8-2
Other ways to create custom web sites for your data	8-2
Static web publishing with HTML	8-2
Using the FileMaker Pro Web Companion	8-2
Requirements for web access	8-2
Enabling the Web Companion	8-3
Setting Web Companion configuration options	8-3
Sharing the database via the Web	8-5

Creating a custom home page	8-5
Specifying a custom home page as the default	8-6
Creating a custom home page for Instant Web Publishing	8-6
About the FileMaker WebPortal object	8-7
Overview of setting up a custom home page for Instant Web Publishing	8-7
Creating a custom web site using a database layout	8-7
Overview of using a database layout as the Instant Web Publishing home page	8-8
Using script buttons in Instant Web Publishing	8-8
Suppressing the Instant Web Publishing interface	8-11
Bypassing the Instant Web Publishing home page	8-12
Format filenames for instant web pages	8-13
Web Companion support for Internet media types	8-13
Monitoring your site	8-14
Using the access.log file	8-14
Using the error.log file	8-14
Using the info.log file	8-15
Using the Web Companion external functions	8-15
Exporting data to a static HTML page	8-16
Testing your site without a network connection	8-17
Opening password-protected databases remotely	8-18
Opening and closing databases using XML	8-18
Opening and closing databases using CDML	8-19

Chapter 9

Custom web publishing using CDML

About the CDML examples	9-1
General steps for custom web publishing using CDML	9-2
About CDML format files	9-2
Generating FileMaker Pro CGI requests using CDML	9-3
Request names	9-4
Requests for adding records to a portal	9-4
Requests for editing multiple records in a portal	9-4

Using the CDML Tool and templates	9-5	Example of XML data in the FMPXMLLAYOUT grammar	10-7
Using the Templates tab	9-6	About UTF-8 encoded data	10-8
Using the Tags tab	9-6	Generating FileMaker Pro CGI requests for an XML document	10-8
Customizing a format file template	9-6	Request and parameter names	10-8
Categories of CDML tags	9-8	Requests for adding records to a portal	10-9
Using an intratag parameter	9-9	Requests for editing multiple records in a portal	10-10
About the CDML Reference database	9-9	Using style sheets with your XML document	10-10
Creating error messages	9-11	Comparing CSS, XSLT, and JavaScript	10-11
Using an encoding parameter with a CDML replacement tag	9-12	Cascading style sheets (CSS) example	10-13
Planning your web site	9-12	Extensible Stylesheet Language–Transformations (XSLT) example	10-14
Looking at the three CDML examples	9-13	JavaScript scripting language example	10-16
Employee Database example	9-14	Looking at the XML Inventory example	10-17
Guest Book example	9-14		
Shopping Cart example	9-15		
Chapter 10		Chapter 11	
<i>Using FileMaker Pro XML to deliver your data on the Web</i>		<i>Using JDBC to deliver your data</i>	
About the XML examples	10-1	About the JDBC examples	11-1
General process for custom web publishing using XML	10-2	About JDBC	11-1
Generating an XML document	10-2	Using the FileMaker JDBC Driver	11-2
About XML namespaces	10-3	About the FileMaker JDBC Driver	11-2
About FileMaker Pro database error codes	10-3	Using a JDBC URL to connect to your database	11-2
Using the FMPDSORESULT grammar	10-3	Specifying driver properties in the URL subname	11-3
Description of elements in the FMPDSORESULT grammar	10-4	SQL supported by the FileMaker JDBC Driver	11-4
Example of XML data in the FMPDSORESULT grammar	10-4	Using DbOpen and DbClose pseudo procedures	11-5
Using the FileMaker Pro Extended XML grammars	10-5	Using the RecordID pseudo column	11-6
Description of elements in the FMPXMLRESULT grammar	10-5	Using the ModID pseudo column	11-6
Example of XML data in the FMPXMLRESULT grammar	10-6	SQL statement examples	11-7
Description of elements in the FMPXMLLAYOUT grammar	10-7	Using a character escape	11-7
		FileMaker data type mapping to JDBC SQL and Java data types	11-7
		FileMaker Pro support for Unicode characters	11-7
		About the FileMaker JDBC Driver interfaces and extensions	11-8

Example 1: Looking at the FileMaker Pro Explorer application	11-8	Customizing FMExample.rc	12-6
Setup requirements	11-9	Customizing FMPrefs.c	12-6
Install the example and the FileMaker JDBC Driver	11-9	Customizing FMMain.h	12-6
Open and share your databases via the Web	11-9	Customizing FMFunc.h	12-6
Run the FileMaker Pro Explorer application	11-9	Customizing FMFunc.c	12-6
View the source code of the example	11-10	Requirements for writing an external function plug-in	12-7
Example 2: Creating the JBuilder Inventory application	11-11	Required code files	12-7
Install the example and FileMaker JDBC Driver	11-11	Required resource files	12-7
Set up JBuilder to use the FileMaker JDBC Driver	11-11	Feature string syntax	12-8
Open and share the Inventory.fp5 database	11-11	Requirements for the plug-in's main entry point	12-9
Start a new JBuilder project	11-11	External function naming conventions	12-10
Create the data module	11-12	FileMaker Pro messages sent to the plug-in	12-10
Design the data module	11-12	The Initialization message	12-11
Test the data module	11-13	The Shutdown message	12-11
Generate the application	11-13	The Idle message	12-11
Example 3: Creating the Visual Cafe Inventory application	11-15	The External Function message	12-12
Install the example and the FileMaker JDBC Driver	11-15	The Preferences message	12-12
Set up Visual Cafe to use the FileMaker JDBC Driver	11-15	Debugging your plug-in	12-13
Open and share the inventory_db database	11-15	Avoiding potential Mac OS resource conflicts	12-13
Create a new Visual Cafe project	11-15	Providing documentation for your plug-in	12-13
		Registering your plug-ins	12-13
		Steps for registering your external function plug-in	12-13
		Revising a registered plug-in	12-14
Chapter 12		Appendix A	
<i>Understanding external function plug-ins</i>		<i>Feature comparison of the runtime application and FileMaker Pro</i>	
About external functions	12-1	Application and document preferences	A-1
About the plug-in example file	12-1	Toolbar comparison	A-3
Contents of the FMExample Plugin folder	12-2	Menu command comparison	A-3
Contents of the EFP API folder	12-2	Ignored script steps	A-6
Contents of the Resources folder	12-2	Stored registry settings	A-6
Contents of the Source folder	12-2		
Installing, enabling, and configuring the example plug-in	12-3		
Description of the FMExample plug-in's external functions	12-4		
Using the example plug-in	12-4		
Customizing the plug-in example	12-6		
Customizing FMExample.r	12-6		

Appendix B***Valid names used in CGI requests for FileMaker Pro XML data***

Generating a <code>-find</code> , <code>-findall</code> , or <code>-findany</code> request	B-1
Examples of <code>-find</code> , <code>-findall</code> , and <code>-findany</code> requests	B-1
Generating a <code>-view</code> request	B-2
Examples of <code>-view</code> requests	B-2
Generating a <code>-new</code> request	B-2
Examples of <code>-new</code> requests	B-2
Generating an <code>-edit</code> request	B-3
Examples of <code>-edit</code> requests	B-3
Generating a <code>-delete</code> request	B-3
Examples of <code>-delete</code> requests	B-3
Generating a <code>-dbnames</code> request	B-3
Examples of <code>-dbnames</code> requests	B-4
Generating a <code>-layoutnames</code> request	B-4
Examples of <code>-layoutnames</code> requests	B-4
Generating a <code>-scriptnames</code> request	B-4
Examples of <code>-scriptnames</code> requests	B-4
Generating a <code>-dbopen</code> request	B-5
Examples of <code>-dbopen</code> requests	B-5
Generating a <code>-dbclose</code> request	B-5
Examples of <code>-dbclose</code> requests	B-5
Generating a <code>-dup</code> request	B-5
Examples of <code>-dup</code> requests	B-6
Generating an <code>-img</code> request	B-6
Examples of <code>-img</code> requests	B-6
Specifying parameters for the request	B-6
<code>-db</code> (Database)	B-6
<code>-lay</code> (Layout)	B-6
<code>-format</code> (Format)	B-7
<code>-recid</code> (Record ID)	B-7
<code>-modid</code> (Modification ID)	B-7
<code>-lop</code> (Logical operator)	B-7

<code>-op</code> (Comparison operator)	B-8
<code>-max</code> (Maximum records)	B-8
<code>-skip</code> (Skip records)	B-8
<code>-sortfield</code> (Sort field)	B-9
<code>-sortorder</code> (Sort order)	B-9
<code>-script</code> (Script)	B-9
<code>-script.prefind</code> (Script before Find)	B-9
<code>-script.presort</code> (Script before Sort)	B-9
<code>-styletype</code> (Style type)	B-10
<code>-stylehref</code> (Style href)	B-10
<code>-password</code> (Database password)	B-10
field name (Name of specific field)	B-10

Appendix C***FileMaker Pro values for error codes***

C-1

Index

I-1

Chapter 1

Welcome to FileMaker Developer

Welcome to FileMaker® Developer 6. FileMaker Developer 6 consists of the Developer edition of FileMaker Pro, and the FileMaker Developer Tool application. You use FileMaker Pro to create and test your database solutions. You use FileMaker Developer Tool to transform your database solution into a runtime or kiosk-mode application that you can distribute to your users. Throughout this Guide, references to FileMaker Pro include the Developer edition of FileMaker Pro.

About the Developer edition of FileMaker Pro

The Developer edition of FileMaker Pro includes two productivity tools designed especially for database developers. They are the Script Debugger tool and the Database Design Report tool. These tools are fully documented in the FileMaker Pro onscreen Help:

- In the Contents tab, see “Using FileMaker Developer features”
- In the Index tab, look for Script Debugger or Database Design Report

Note FileMaker Inc. does not support the use of the Script Debugger and Database Design Report tools on files processed as runtime applications through the FileMaker Developer Tool.

About this guide

This *Guide* tells you how to install FileMaker Developer on your Windows or Mac OS computer and how to use the FileMaker Developer Tool. In addition, there are several chapters that explain how to publish a FileMaker Pro database on the Web.

Note You cannot publish a runtime solution on the Web.

The FileMaker Pro *User’s Guide*, also included in the FileMaker Developer package, describes how to use the most common features in FileMaker Pro. For a list of new features in FileMaker Pro 6, see chapter 4 of the FileMaker Pro *Getting Started Guide*, or see “About FileMaker Pro 6” in the onscreen Help.

Note When a feature or procedure is specific to a particular platform, you see instructions and illustrations that are also specific to that platform. For features or procedures that are similar on both platforms, you may see illustrations for either Windows or the Mac OS.

Registration and customer support

Please take the time to register your product during installation, through the FileMaker web site at www.filemaker.com/register, or by choosing Help menu > Register Now in FileMaker Pro.

For information about technical support and customer service, see:
www.filemaker.com (North American customers)
www.filemaker.com/intl (customers outside North America)

Or choose Help menu > FileMaker on the Web. At the web site, you will find the FileMaker Service Directory, which details the service options available to North American customers, as well as links to FileMaker international sites, answers to frequently asked questions, and access to the TechInfo knowledgebase used by Technical Support. If you do not have access to the Web, please refer to the FileMaker Service Directory included in the software box. North America customers can also call 1-800-965-9090 to learn about the service options available.

About the installation code

The installation code is a seventeen-digit number located on a multi-part sticker on a separate paper sheet within the box. Do not lose this installation code; it cannot be replaced. We recommend that you place one of the stickers on the inside front cover of your manual.

Important You must enter the installation code during installation or the software will not install.

The installation code ensures adherence to the single user license agreement, which allows for use of one (1) copy of the Software on a single computer at a time (refer to your Software License). If the code is invalid or if another copy of the application installed with that same code is running on the network, the application displays this error message: “The maximum number of licensed users are currently using this copy of FileMaker Pro. Please refer to the Installation Code section of your Getting Started Guide for further instructions.”

If you receive this error message, you have entered a duplicate installation code. To install FileMaker Pro on multiple computers, you must have a unique installation code for each user, or obtain a volume license. You must license one copy of FileMaker Pro for each computer.

Hardware and software requirements

To install and use FileMaker Developer, you need the CD-ROM in the FileMaker Developer package and the following minimum equipment and software:

Windows requirements

- an Intel-compatible Pentium 90 or higher
- at least 32 MB of RAM
- a hard disk with at least 210 MB of free space (for a Complete installation)
- a CD or DVD drive
- Windows 98, Windows Me, Windows XP, Windows 2000 Professional, or Windows NT 4.0 with Service Pack 6

Mac OS requirements

- an Apple computer that supports and is running Mac OS 8.6, Mac OS 9 versions 9.0 through 9.2.2, or Mac OS X versions 10.1 through 10.1.4
- a CD or DVD drive
- the amount of RAM and disk space required depends on the Mac OS version:

	RAM	Disk space (for an Easy Install)
Mac OS 8.6 or Mac OS 9.x	at least 32 MB	at least 60 MB
Mac OS X	at least 128 MB	at least 60 MB

Important note about supported operating systems

At the time this book was written, FileMaker, Inc. tested FileMaker Pro with the Windows and Mac OS operating systems listed above. FileMaker Pro may or may not work with newer operating system releases. For information on newer operating systems, see www.filemaker.com.

Some FileMaker Pro features are unique to or work differently on certain operating systems. This manual and online Help use the label “Mac OS 9.x or earlier” to indicate when a described feature is specific to Mac OS 8.6 through 9.2.2. However, keep in mind that Mac OS 8.6 is the earliest Mac OS version that FileMaker Pro supports.

Networking requirements

If you plan to share FileMaker Pro files with other computers on a network, you need to know which type of network you’re using. FileMaker Pro supports the following network protocols:

FileMaker supports this network protocol	On these operating systems
TCP/IP (recommended)	All supported Windows and Mac OS operating systems
IPX/SPX	Supported Windows operating systems only
AppleTalk	Mac OS 9.x or earlier

If you’re not sure which network protocol is installed and in use on your computer, check with your network administrator before you begin installation.

Note FileMaker recommends using the TCP/IP protocol on both Windows and Mac OS networks. You must use the TCP/IP protocol to share files on a mixed Windows and Mac OS network.

Computers running Windows also require a Windows-compatible network card and a software driver for the card.

Web publishing requirements

To publish a database on the Web using the FileMaker Pro Web Companion, you need a host computer with access to the Internet or an intranet via TCP/IP. The Web Companion network protocol (TCP/IP) is independent from the network protocol you choose for FileMaker Pro. For example, you can choose AppleTalk or IPX/SPX for the FileMaker Pro network protocol and still publish your database on the Web via TCP/IP.

Important Access to the Internet requires an Internet service provider (ISP). FileMaker Pro does not provide an Internet account for you.

Requirements for advanced features (Windows)

Some of the advanced features of FileMaker Pro require additional software.

FileMaker Pro advanced feature	Software requirement
Send Mail script step	A Mail API (MAPI) enabled e-mail client software
Phone dialing	Phone dialer or other Telephony API (TAPI) compliant software
ODBC features	Microsoft Data Access Components (MDAC) version 2.5. To update MDAC software, use Microsoft Windows Update or see the Microsoft web site at www.microsoft.com .
Insert QuickTime movie and image files	QuickTime software (available at www.apple.com)

Requirements for advanced features (Mac OS)

FileMaker Pro advanced feature	Software requirement
Send Mail script step	E-mail client software
Insert QuickTime movie and image files	QuickTime software (available at www.apple.com if it was not included with your Mac OS version)

FileMaker Pro advanced feature	Software requirement
URL support and the Speak script step	In Mac OS 9.x or earlier, the Internet control panel and the Speech Manager extension are usually available as part of a normal system software installation. Make sure they're turned on in the Extensions Manager control panel.

Features not supported in Mac OS X

In Mac OS X, FileMaker Pro does not support the following features:

- Toolbars
- Status (CurrentPrinterName) function
- Dial Phone script step

Opening files from previous versions

FileMaker Pro 6 can open files created in earlier versions of FileMaker Pro as described in the following sections.

Important There may be date considerations when converting a database from an earlier version of FileMaker Pro. For more information about how FileMaker Pro handles Year 2000 issues, see www.filemaker.com.

FileMaker Pro 5.x databases

Because FileMaker Pro 5.x and FileMaker Pro 6 share the same format, FileMaker Pro 6 can open FileMaker Pro 5.x files without converting them. You can even use FileMaker Pro 6 files with FileMaker Pro 5.x. However, the new features that you use in a newer version will not be supported when you open the file in a previous version.

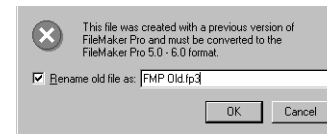
We recommend that once you have created or opened a file in FileMaker Pro 6, you do not make database design changes in previous versions of the application, particularly to features that have changed in FileMaker Pro 6.

FileMaker Pro 4.x/3.x/2.x databases

When you convert a FileMaker Pro 4.x/3.x/2.x file, FileMaker Pro 6 software saves your original FileMaker Pro 4.x/3.x/2.x file and creates a copy that has been converted to the FileMaker Pro 5.x/6 format. The original file is not modified and you can open it in the previous version of FileMaker Pro. The converted file can only be opened in FileMaker Pro 5.x/6.

To convert and open a FileMaker Pro 4.x/3.x/2.x file:

1. Start FileMaker Pro 6.
2. Choose File menu > Open and select the file to convert.
3. Click Open.
4. Click OK to append “Old” to the name of the original FileMaker Pro 4.x/3.x/2.x file.



5. Click Save to start the conversion.

By default, this converted file will have the original file's name. Preserving this name is important if you have existing relationships or scripts, which might not function correctly if it is changed.

FileMaker Pro converts the file and opens it.

Keep these points in mind when converting files:

- You can also convert a FileMaker Pro file by dragging the file on top of the FileMaker Developer 6 application icon.

- If you add or remove the .FP5 filename extension as part of the conversion process, you must re-specify file locations for related or external files when the database is first opened.
- If you are converting a copy of a FileMaker Pro file, close the file before you copy it. Files that are copies of open files will not convert correctly.
- If consistency check or auto-repair error messages appear during conversion, try recovering the file first using a previous version of FileMaker Pro.

FileMaker Pro 1.x databases

On the Macintosh, you can convert FileMaker Pro 1.x databases with FileMaker Pro 6. Follow the instructions in the previous section, “FileMaker Pro 4.x/3.x/2.x databases,” to convert your file to FileMaker Pro 6.

FileMaker Pro for Windows cannot convert FileMaker Pro 1.x databases directly. To use a database created by FileMaker Pro 1.x with FileMaker Pro 6 on Windows, do the following:

1. Convert the file on a Mac OS computer using FileMaker Pro 2.x or later.

If you have a Mac OS computer but do not have FileMaker Pro 2.x or later, you can download a trial version of FileMaker Pro for Macintosh from www.filemaker.com, and use it to convert the file.

2. Transfer the converted file to your Windows-based computer.
3. If necessary, follow the instructions in the previous section, “FileMaker Pro 4.x/3.x/2.x databases,” to convert the file to FileMaker Pro 6.

Updating your User Dictionary

If you added words to a User Dictionary in a previous version of FileMaker Pro, you may want to use it with FileMaker Pro 6. For more information about converting a User Dictionary, see FileMaker Pro Help after you have installed FileMaker Developer 6. Choose Help menu > Contents and Index, click the Index tab, and type dictionaries.

Read Me file

In the FileMaker Developer 6 folder, the Read Me file contains last-minute information about FileMaker Developer 6.

Electronic documentation

In the Electronic Documentation folder in the installed FileMaker folder and on the CD you will find PDF versions of the following documents:

Filename	Document title
FMD 6 Developer’s Guide.pdf	FileMaker Developer 6 Developer’s Guide
FMP 6 User’s Guide.pdf	FileMaker Pro User’s Guide
FMP 6 Getting Started.pdf	FileMaker Pro 6 Getting Started Guide
Customizing Templates.pdf	Customizing your FileMaker Pro templates

In addition, FileMaker Pro 6 installs the Web Security.pdf in the FileMaker Pro 6\Web Security folder on your hard drive.

Abiding by the license agreement

The FileMaker Developer license agreement allows you royalty-free distribution of an unlimited number of FileMaker Pro runtime database solutions. However, there are several terms and conditions in the license agreement you must abide by, including the following:

- You must provide all of the end-user technical support.
- You must provide an “About” layout that includes your name, address, and the telephone number for your technical support. For more information, see “Creating an About layout” on page 4-8.

Any end users to whom you distribute the ODBC driver will also be subject to the ODBC driver software license agreement provided with the ODBC driver.

Note You must read and agree to the terms and conditions of the FileMaker Developer license agreement, available through the FileMaker Developer installer, before using the FileMaker Developer software.

About the TechInfo database

The TechInfo Knowledgebase is a great resource for technical information about FileMaker, Inc. products. This FileMaker Pro database serves as a front-line resource for the company’s Technical Support staff as they field customer inquiries. It is a collection of Q&As, tips, FAQs, issue reports, update notes, press releases, and a host of other material valuable for the support professional.

The TechInfo Knowledgebase is available on the product support pages of the FileMaker, Inc. web site at www.filemaker.com.

Chapter 2

Installing FileMaker Developer in Windows

Before you begin the installation process, exit other open programs and save your work. For late-breaking information about FileMaker Developer, browse the contents of the CD and view the Read Me file or visit www.filemaker.com.

Installation notes

Before you install FileMaker Developer, review the following notes:

- To install FileMaker Developer in Windows XP or Windows 2000 Professional, you must log on with Administrator or Power User privileges. Windows NT users must log on with Administrator privileges.
- If FileMaker Developer 5.x is already installed, you can install FileMaker Developer 6 without uninstalling FileMaker Developer 5.x. You can use either version of FileMaker Developer on the same computer.
- If your computer is running virus protection software, turn it off prior to installing FileMaker Developer, and then turn it back on afterwards.
- When you are installing FileMaker Developer, you will see the License Agreement panel. Read the Software License Agreement. If you accept the terms of the license agreement, click **I accept...**, then click **Next**.
- By default, FileMaker Developer 6 will be installed in the C:\Program Files\FileMaker\FileMaker Developer 6 folder. During installation, you can choose another drive or another folder if you wish.

- During installation, you can choose whether to install all program features (**Complete**), or to install only selected portions of the program (**Custom**). For example, you may choose not to install certain features in order to save disk space.
- During installation, you will be prompted to choose one of the following network protocols for sharing databases: **TCP/IP**, **IPX/SPX**, or **None**. If you are not certain which protocol to use, ask your network administrator, or choose **TCP/IP**. (If you make a mistake and choose the wrong network protocol, you can change it later. See “Adjusting network software” on page 2-3 after you have installed FileMaker Developer.)

Installing FileMaker Developer

You must use the installation program to install FileMaker Developer—you can’t install FileMaker Developer by dragging files to your hard disk.

If you want to create an install log file that lists the registry entries and paths of all installed files, you must install FileMaker Developer 6 from the command line. See “Installing FileMaker Developer from the command line” on page 2-2.

To install FileMaker Developer on your hard disk:

1. Insert the CD into your CD or DVD drive.
2. Click **Install FileMaker Developer 6**.



The FileMaker Developer InstallShield Wizard appears.

3. Install FileMaker Developer by following the on-screen instructions.

For details about installation issues, see “Installation notes” on page 2-1.

4. When the installation is complete, click Finish.

When installation completes, you see a Product Registration panel. You can register online if you have an Internet connection. If you choose to register by letter or by fax, a form will appear in the Adobe Acrobat Reader application. You can fill out the form on your computer, print a copy of the form, and mail or fax it to FileMaker.

Installing FileMaker Developer from the command line

To install FileMaker Pro from the command line and create an installation log file:

- 1.** Insert the FileMaker Developer 6 CD.
- 2.** Open a command window.

3. In the command window, type

```
E:\Files\setup.exe /v"/Lr C:\FMP6InstallLog.txt"
```

4. Follow the on-screen instructions.

The FMP6InstallLog.txt file will be located on the C drive of your hard disk. You can view it with Notepad, Wordpad, or a word processing program.

Note If your CD or DVD drive is not located at drive E, then substitute the correct letter for your drive.

Where files are stored on your hard disk

The FileMaker Pro installer copies files to several folders on your hard disk, depending on the installation options you choose and your operating system. If you performed a command line installation and specified an install log file, you can view the install log file which lists the location of each installed file. This file can be viewed with Notepad, WordPad or a word processing application.

Uninstalling, modifying, or repairing FileMaker Developer

You can use the Windows Add/Remove Programs Control Panel to uninstall FileMaker Developer, replace missing or corrupt files, or to change which features are installed. For more information about using Add/Remove Programs, refer to Windows Help.

Note Uninstalling doesn't delete files that you've created, as long as they are named differently than the sample and template files that are installed with FileMaker Developer.

When you modify or repair FileMaker Developer, you may be prompted to enter your installation code.

Adjusting network software

Depending on the networking protocol you're using, you may have to modify settings on the host and/or guest computers.

Changing the network protocol

During installation, you select a network protocol (or none) for FileMaker Pro. You must choose a network protocol if you want to share FileMaker Pro databases over a network, either as a guest or as a host. The protocol you choose for FileMaker Pro database sharing does not affect other intranet or Internet activities. You can change the FileMaker Pro database sharing network protocol at any time without reinstalling the software.

Important The network protocol setting in FileMaker Pro must be the same on the host and guest computers.

For more information about your network configuration, see “Networking requirements” on page 1-3, or consult your network administrator.

To change the network protocol:

1. In FileMaker Pro, choose Edit menu > Preferences > Application.
2. In the General tab, choose a network protocol from the Network Protocol drop-down list.
3. Click OK.

Changes to this setting don't take effect until you restart FileMaker Pro.

Changing the IPX/SPX frame type

If you're using IPX/SPX as the network protocol, you must select a frame type. A frame encapsulates packets of information in a recognizable format; all computers communicating with each other on a network must use the same frame type. For help in deciding which frame type to select, see your network administrator.

Note If you are using TCP/IP networking, you do not need to change the frame type.

Changing the frame type (Windows 98, and Windows Me)

To change the frame type with IPX/SPX networking:

1. Open the Network control panel and click the Configuration tab.
2. Select the IPX/SPX protocol from the list and click Properties.
3. Click the Advanced tab.
4. Click the frame type in the Property dialog box.
5. Make a selection from the Value drop-down list.

Note To allow the maximum recommended number of guests (10) to connect, consider increasing the IPX/SPX Maximum Connections to 90 or greater in the Network control panel.

6. Click OK.

Windows prompts you to restart your computer for these new changes to take effect.

Changing the frame type (Windows NT 4.0)

To change the frame type with IPX/SPX networking:

1. Open the Network control panel and click the Properties tab.
2. Select the IPX/SPX protocol from the list and click Properties.
3. Select a frame type from the drop-down list.
4. Click OK to apply the frame type, and click OK to close the Network dialog box.

Windows prompts you to restart your computer for these new changes to take effect.

Changing the frame type (Windows XP and Windows 2000 Professional)

To change the frame type with IPX/SPX networking:

1. From the Start menu, open the appropriate control panel:

- Windows XP: the Network Connections control panel
 - Windows 2000 Professional: the Network and Dial-up Connections control panel
2. Right-click on the network connection you are using.
 3. Select Properties from the menu.
 4. Select the IPX/SPX protocol from the list and click Properties.
 5. Select a frame type from the drop-down list.
 6. Click OK to apply the frame type.
 7. Click OK to close the Connection Properties dialog box.
 8. Close the Network and Dial-up Connections window.

With Windows XP and Windows 2000 Professional, you do not need to restart your computer.

Changing the FileMaker Pro cache size

FileMaker Pro maintains an internal cache of portions of your database. Depending on your computer's memory configuration, you may want to adjust the cache size to improve performance.

You may want to consider increasing the cache size if you plan to:

- work on a large database file
- work on several database files at once
- host database files
- use a database file that contains many graphics
- insert a large graphic into a database file

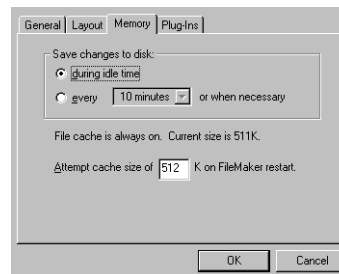
Consider decreasing the cache size if you plan to run several applications at the same time and you have a limited amount of memory (RAM) installed.

Note When you increase or decrease the FileMaker Pro cache size, you also increase or decrease the amount of memory needed to run FileMaker Pro.

For more information on managing memory, see the documentation that came with your computer.

To change the cache size that FileMaker Pro uses:

1. In FileMaker Pro, choose Edit menu > Preferences > Application.
2. Click the Memory tab.



3. Type a number in the Attempt cache size of box.
4. Click OK.

The next time you start FileMaker Pro, the new cache size is used.

Where to go from here

After you install FileMaker Developer, you can immediately begin working with the application. Here are some suggestions on where to go to get started:

- If you're new to databases, read chapter 1, "FileMaker Pro basics," in the *FileMaker Pro User's Guide* to learn basic database concepts.
- If you're new to FileMaker Pro, work through the tutorial in the *FileMaker Pro Getting Started Guide*. The tutorial lesson files are automatically installed when you do a Complete installation.
- If you've used FileMaker Pro before, read chapter 4 in the *FileMaker Pro Getting Started Guide*, "New features in FileMaker Pro," to learn about new features available in FileMaker Pro 6.

Chapter 3

Installing FileMaker Developer in the Mac OS

Before you begin the installation process, quit other open programs and save your work. For late-breaking information about FileMaker Developer, browse the contents of the CD and view the Read Me file or visit www.filemaker.com.

Installation notes

Before you install FileMaker Developer, review the following notes:

- The FileMaker Developer CD contains two installer programs, one for Mac OS 9.x or earlier, and one for Mac OS X. Before you install FileMaker Developer, confirm your Mac OS version. With the Finder application active, choose **About this Computer** or **About this Mac** from the Apple menu. A window will appear that lists your Mac OS version.
- If you're upgrading, you can install FileMaker Developer 6 without removing the previous version of FileMaker Developer that you may already have installed.
- If your computer is running virus protection software, turn it off prior to installing FileMaker Developer, and then turn it back on afterwards.
- During installation, you can choose whether to install all the program files (an Easy Install), or to install portions of the program (a Custom Install). For example, you may choose not to install certain features in order to save disk space.
- Mac OS X: if you're installing FileMaker Developer on a computer with multiple user accounts, see "Accessing files from other user accounts (Mac OS X)" on page 3-2 after you install FileMaker Developer.

- Mac OS X: ODBC drivers are not automatically installed. You must install the drivers manually. See "Installing ODBC drivers in Mac OS X" on page 3-2.

Installing FileMaker Developer

You must use the installation program to install FileMaker Developer—you can't install FileMaker Developer by dragging files to your hard disk.

To install FileMaker Developer on your hard disk:

1. Insert the CD into your CD or DVD drive.

You see the disc's window on your screen. If you see only the disc icon, double-click the CD icon to open its window.

2. Using the table below, locate and double-click the appropriate **Start Here** file for the Mac OS version in use on your computer.

For this Mac OS version: Double-click this file:

Mac OS 9.x or earlier



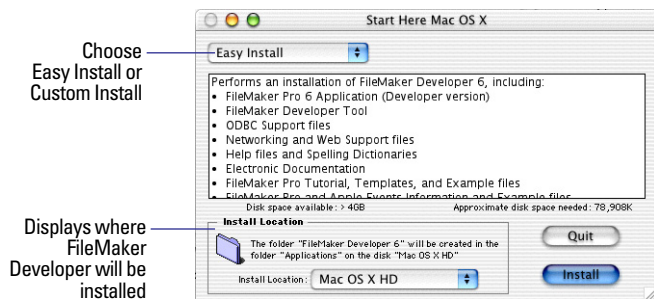
Mac OS X



You see the Software License Agreement.

3. Read the Software License Agreement. If you accept the terms of the license agreement, click **Accept**.

You see the FileMaker Developer Installer dialog box.



4. Choose Easy Install or Custom Install from the pop-up menu.

The Easy Install option installs all of the FileMaker Developer program files. The Custom Install option allows you to choose which portions of the product to install. (In most cases you should use the Easy Install option.)

5. Look in the Install Location area, and make a note of the folder and hard drive where FileMaker Developer will be installed.

If you wish, you can choose another hard drive or another folder from the Install Location pop-up menu.

6. Click Install.

You see the personalization dialog box.

7. Type your name, company name (if any), and your FileMaker Developer installation code into the appropriate text boxes.

The installation code is a seventeen-digit number located on a multi-part sticker on a separate paper sheet within the box. See “About the installation code” on page 1-2 for more information.

8. Click OK.

The Installer begins copying files.

When installation completes, you see a Product Registration dialog box. You can register online if you have an Internet connection. If you choose to register by letter or by fax, a form will appear in the Adobe Acrobat Reader application. You can fill out the form on your computer, print a copy of the form, and mail or fax it to FileMaker.

A final dialog box tells you the installation on your hard disk was successful.

9. Click Quit to leave the Installer.

Installing ODBC drivers in Mac OS X

In Mac OS X, you must manually install the ODBC drivers after the FileMaker Developer installation is complete.

1. Navigate to the folder Applications\FileMaker Developer 6\FileMaker ODBC Drivers.
2. Double-click the FileMaker Pro ODBC Drivers file.
3. Follow the installer instructions.

Note To install the drivers, you will need an administrator password, such as the password created when you first set up Mac OS X.

Where files are stored on your hard disk

The FileMaker Developer installer copies files to several folders on your hard disk, depending on the installation options you choose and your operating system. When installation is complete, you can view the Installer Log file, which lists the location of each installed file.

The Installer Log file is located in the folder where FileMaker Developer was installed. You can view it with TextEdit (Mac OS X), SimpleText (Mac OS 9.x or earlier), or a word processing program.

Accessing files from other user accounts (Mac OS X)

If you logged in as an administrator and installed FileMaker Developer into the default Applications folder on a computer with multiple user accounts, other users who log in with a non-administrator password will not be able to access some FileMaker Developer application folders and files.

- **Template and tutorial files:** If a non-administrator user needs to access these files, perform a Custom Install and install these files into the user's Applications folder.
- **Web Security folder and Web folder:** Set file permissions so that all valid users have Read & Write access to the Web Security databases and Read only access to the Web folder.

For more information on setting file permissions, see the Mac OS X topic "Setting Access Priviledges."

Choosing a network protocol (Mac OS 9.x or earlier)

You must choose a network protocol if you want to use FileMaker Pro over a network, either as a guest or as a host. For more information about your network, see "Networking requirements" on page 1-3, or consult your network administrator.

Important The network protocol setting in FileMaker Pro must be the same on the host and guest computers.

To choose a network protocol:

1. Choose Edit menu > Preferences > Application.
2. In the General tab, choose a network protocol from the Network protocol pop-up menu.
3. Click OK.

Changes to this setting don't take effect until you restart FileMaker Pro.

Adjusting memory

You might need to change the amount of memory reserved for FileMaker Pro if you plan to:

- work on a large database file
- work on several database files at once

- host database files
- use a database file that contains many graphics
- insert a large graphic into a database file

Mac OS X allocates memory to applications differently than Mac OS 9.x or earlier. To adjust the memory amount, follow the appropriate instructions for your Mac OS version:

- For Mac OS X, follow the instructions in the next section, "Adjusting the FileMaker Pro cache (Mac OS X)."
- For Mac OS 9.x or earlier, see "Adjusting memory requirements (Mac OS 9.x or earlier)" on page 3-4.

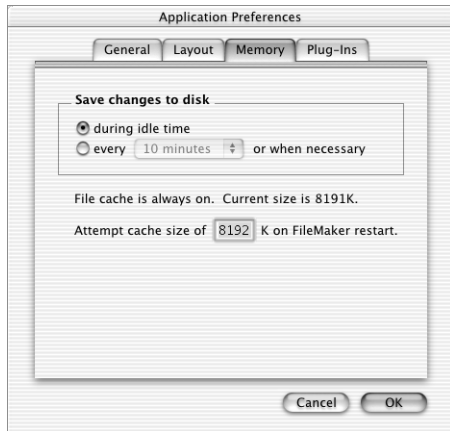
Adjusting the FileMaker Pro cache (Mac OS X)

Because Mac OS X dynamically allocates memory to applications as needed, there is no way to set a specific amount of memory to allocate to the FileMaker Pro application. However, you can adjust the FileMaker Pro cache size.

You may want to increase the cache size if you plan to do any of the items listed in "Adjusting memory," above. Consider decreasing the cache size if you plan to run several applications at the same time and you have a limited amount of memory (RAM) installed.

To change the cache size that FileMaker Pro uses:

1. In FileMaker Pro, choose the FileMaker Developer application menu > Preferences > Application.
2. In the Application Preferences dialog box, click the Memory tab.



3. Type a number in the Attempt cache size of box.

4. Click OK.

The next time you start FileMaker Pro, the new cache size is used.

Adjusting memory requirements (Mac OS 9.x or earlier)

You may want to increase the amount of memory reserved for FileMaker Pro if you plan to do any of the items listed in “Adjusting memory” on page 3-3. Consider decreasing the amount if you plan to run several applications at the same time and you have a limited amount of memory (RAM) installed.

Keep the following points in mind:

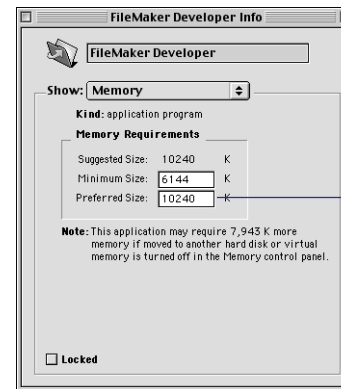
- The amount of memory you give to FileMaker Pro affects your ability to run other applications. If the Mac OS and FileMaker Pro take up all the available memory, you won't be able to open other applications while FileMaker Pro is running.
- Avoid specifying less memory than the Suggested size in the FileMaker Developer Info dialog box.

To change the amount of memory FileMaker Pro uses:

1. If FileMaker Pro is open, quit the application by choosing File menu > Quit.
2. In the Finder software, open the FileMaker Developer 6 folder. Select the FileMaker Developer application icon by clicking it once.

Note If you click the application icon twice, you will start the Developer edition of FileMaker Pro. If that happens, choose File menu > Quit and try again.

3. Choose File menu > Get Info > Memory.



Change the amount of memory FileMaker Pro uses here

4. In the Memory Requirements area, double-click the Preferred Size number and type a number to change the memory allocated for FileMaker Developer.

If you can't type in the box, make sure you've quit the Developer edition of FileMaker Pro.

5. Click the close box.

The next time you start the Developer edition of FileMaker Pro, it will use the amount of memory you specified in the Preferred Size text box if that amount of memory is available.

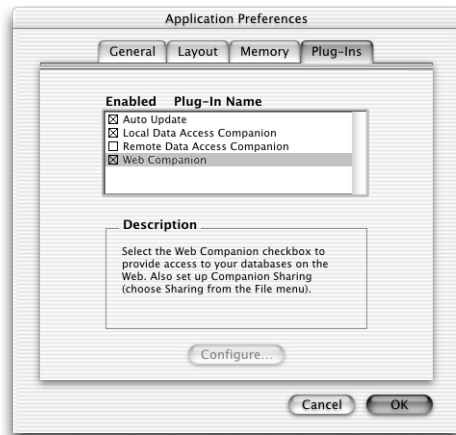
Enabling the FileMaker Pro Web Companion (Mac OS X)

Note For instructions on enabling the FileMaker Pro Web Companion in Mac OS 9.x or earlier, see chapter 14 of the *FileMaker Pro User's Guide*.

The Developer edition of FileMaker Pro uses the FileMaker Pro Web Companion plug-in to serve databases over the Web.

To enable the Web Companion in Mac OS X:

1. Start the FileMaker Developer application.
2. Choose the FileMaker Developer application menu > Preferences > Application.
3. In the Application Preferences dialog, click the Plug-Ins tab.
4. Select the Web Companion checkbox.



After you enable the Web Companion, you must specify which *port number*, or virtual connection, the Web Companion will use to publish data.

The first time you enable the Web Companion, FileMaker Pro displays the following dialog, which requests permission to make a one-time change to your computer's setting to facilitate web publishing on a port number below 1024.



The standard port number for web publishing is 80 (ports are numbered between 1 and 65535), and most web servers and browsers use this port as the default. Port 80 is also the default port for the FileMaker Pro Web Companion. For security reasons, Mac OS X restricts access to ports below 1024. To configure the FileMaker Pro Web Companion to use ports below 1024 while maintaining the Mac OS X access restrictions on these ports, it is necessary to make a one-time change to the file permissions of the Web Companion Enabler to give it the authority to open privileged ports (ports 1-1023). To make this change, you will need an administrator password, such as the password created when you first set up Mac OS X.

Note If you use a port other than port number 80 for FileMaker Pro web publishing, your users will need to append a colon (":") and the number of the port to their URLs to access your web hosted databases. For more information, see "Accessing databases that are published to the Web" on page 3-7.

5. Do one of the following three options:

To	Click
Configure the Web Companion to use the default port 80, or any other valid port number (This is the recommended option, but it requires a Mac OS X administrator password.)	Continue. Proceed with the next step in this section.
Configure the Web Companion to use port 1024 or higher (This option does not require a Mac OS X administrator password.)	Advanced. For further instructions, see the section “Configuring the Web Companion for use with ports 1024 and higher” on page 3-6.
Skip enabling the Web Companion at this time	Cancel. The Web Companion will not be enabled, and your system settings remain unchanged.

6. Enter an administrator name and password in the Authenticate dialog.

The administrator name and password you enter can be the same as the name and password used when Mac OS X was installed, or if you have administrator privileges but do not know an administrator password, you can create a new user and password with administrator privileges. For more information on creating an account with administrator privileges, see the Mac OS X Help topics “Working as an administrator” and “Changing your password.”



7. Click OK.

The Web Companion is now configured to use port 80.

8. Click OK again to close the Application Preferences dialog.

To use a port number other than 80, open the Application Preferences dialog again (choose the FileMaker Developer application menu > Preferences > Application), click the Plug-Ins tab, select the Web Companion plug-in, and click Configure. In the Web Companion Configuration dialog, enter the port number you want (between 1 and 65535) in the TCP/IP Port Number box, and click OK.

Configuring the Web Companion for use with ports 1024 and higher

You do not need an administrator password to configure the FileMaker Pro Web Companion to use a port number between 1024 and 65535. Unlike ports below 1024, the FileMaker Pro Web Companion can use ports 1024 and above without altering your system's settings.

To configure the FileMaker Pro Web Companion to use only ports 1024 and above:

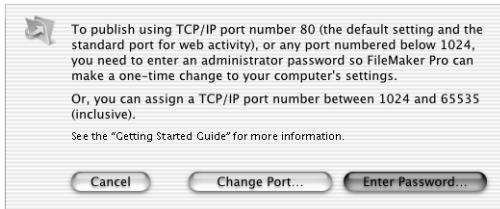
1. Choose the FileMaker Developer application menu > Preferences > Application.
2. In the Application Preferences dialog, click the Plug-Ins tab.
3. Select the Web Companion checkbox.

You see the following dialog.



4. Click Advanced.

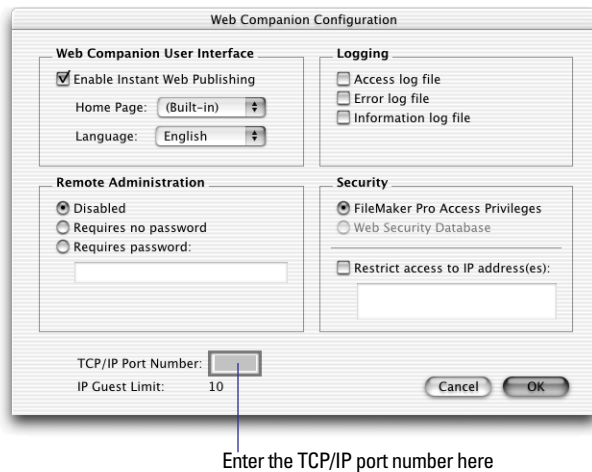
You see the following dialog.



5. Click Change Port.

You see the Web Companion Configuration dialog.

6. Enter a port number between 1024 and 65535 in the TCP/IP Port Number box.



7. Click OK to save your changes.

You are finished. The Web Companion is now configured to use the port you have specified.

Accessing databases that are published to the Web

When you publish a database to the Web, your users access that database by entering the host machine's URL in their web browser. If the FileMaker Pro Web Companion is configured to use port number 80, the default port, the URL your users will enter will look like this:

`http://12.34.56.78/`

The number "12.34.56.78" is an example; in its place, your users would enter the actual URL of your host machine.

If the FileMaker Pro Web Companion is configured to use a port number other than the default, the URL your users will enter will look like this:

`http://12.34.56.78:1024`

Again, the number "12.34.56.78:1024" is an example; your users would enter the actual URL of the host machine, followed by a colon (":") and the port number specified in the FileMaker Pro Web Companion Configuration dialog.

For more information on accessing FileMaker Pro databases on the Web, see chapter 14 of the *User's Guide*, "Publishing databases on the Web."

Where to go from here

After you install FileMaker Developer, you can immediately begin working with the application. Here are some suggestions on where to go in the documentation to get started:

- If you're new to databases, read chapter 1, "FileMaker Pro basics," in the *FileMaker Pro User's Guide* to learn basic database concepts.
- If you're new to FileMaker Pro, work through the tutorial. The tutorial lesson files are automatically installed when you do an Easy Install.
- If you've used FileMaker Pro before, read chapter 4 of the *FileMaker Pro Getting Started Guide* to learn about new features available in FileMaker Pro 6.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 4

Creating a database solution

You use FileMaker Pro to design and build your database solution, and FileMaker Developer Tool to bind the Kiosk-mode or customized stand-alone runtime application.

Note The Developer edition of FileMaker Pro includes Database Design Report and Debug Script menu commands to assist you in designing your database solution. With the Database Design Report feature, you can create schema reports of all your databases, in either a FileMaker Pro database format or in an XML format. With the Script Debugger, you can easily set and clear breakpoints, single-step through your scripts, and run subscripts. For information about using Database Design Report and Script Debugger, see the FileMaker Pro onscreen Help. For information about the Database Design Report XML grammar, see the FileMaker web site at www.filemaker.com.

Before you begin to build your database solution, you need to decide how users will interact with it. Your database solution might have any of the following components:

- a primary database file that connects all of the auxiliary files
- scripts and buttons to open and close auxiliary files, return to the primary file, display a splash screen layout at startup, or quit a runtime application
- common elements and a consistent appearance for cross-platform solutions
- a custom layout theme used for every file in the solution
- an About layout to introduce your solution
- a custom Help system that provides usage tips for your solution
- passwords to protect your databases

The *FileMaker Developer* CD provides an example of a relational database solution. It includes a primary file, named Main Menu.fp5, that contains navigation buttons to several auxiliary files and layouts. Each auxiliary file and layout contains a Main Menu button to return to the Main Menu layout in the primary file.

If you were to prepare these files for a FileMaker Pro runtime database solution, you would need to add an About layout to the primary file, create a script to display the layout, and add a button that quits the runtime application. (See “Considerations for a runtime database solution” on page 4-2 and “Creating an About layout” on page 4-8.)



Main Menu layout of the primary file in the Relational Example

The Main Menu.fp5 file is in the Developer Extras folder:
\\Developer Extras\FileMaker, Inc\Examples\Relational

Overview of preparing your solution files

As you design, build, and test your database solution, keep in mind how your users will interact with it. This includes navigational scripts and buttons, effective use of layouts and themes, and Help for the user.

Here are some general considerations for preparing solution files:

- 1.** If desired, create a custom theme for all the layouts in your database solution. (See chapter 5, “Creating custom layout themes” for information.)
- 2.** In FileMaker Pro, design, create, and test the databases that make up your solution.

If you’re creating a cross-platform solution for Windows and Mac OS machines, design and test each database file on both platforms. See “Design tips for cross-platform solutions” on page 4-11.
- 3.** If necessary, convert files from versions of FileMaker Pro earlier than 5.0.
- 4.** Decide which database will be the primary file that users open first.
- 5.** Create scripts and/or buttons for users to navigate from the primary file to all auxiliary files and layouts in the solution. See “Using scripts to control your solution” on page 4-5 and “Creating dynamic buttons” on page 4-6.
- 6.** Create documentation about your database solution, such as an About layout that describes your company and where users can go for technical support, and a Help layout that describes how to navigate and use your solution. See “Providing user documentation” on page 4-8.
- 7.** Make backup copies of your original files and store them in a safe place.

Once you have completed the design and development work and you are satisfied that your database solution is ready for distribution, use the FileMaker Developer Tool to create Kiosk-mode or custom stand-alone solutions.

To create a custom solution:

- 1.** Use the FileMaker Developer Tool to bind and customize the database files. (See chapter 6, “Using the FileMaker Developer Tool” for information.)
- 2.** Test your custom solution for errors—step through your database files trying every option, feature, and button. If your solution will be run in both Windows and the Mac OS, test your solution on both platforms.
- 3.** Plan how your database solution will be distributed to your users and, if necessary, create a custom installer.

FileMaker Pro runtime applications don’t have automatic installers, so you’ll need to create an installation procedure for your runtime database solution files. You may also need a utility to decompress and install the files on users’ hard disks. See chapter 7, “Distributing FileMaker Pro runtime database solutions” for information.

Considerations for a runtime database solution

Here’s a list of issues to consider as you prepare your database files for a stand-alone runtime database solution. It’s best to address these issues before binding the files to the runtime application.

- Do you want users to open and close auxiliary files?

In the runtime application, several FileMaker Pro menu options are not available to the user, including File menu > Open and File menu > Close. If you want users to open and close auxiliary files, you must provide scripts in your solution to perform these operations. Then, place a button or startup script in each auxiliary file that returns to the main layout of the primary file.

The \Developer Extras\FileMaker, Inc\Examples\Creating Dynamic Buttons\ folder on the *FileMaker Developer* CD contains information about buttons you can use to make opening and closing files easier for users. See “Creating dynamic buttons” on page 4-6 for information.

- Will the runtime application operate in Kiosk mode?

If your runtime database solution will display in Kiosk mode, the entire interface must be accessible via buttons on the layouts. See “Creating dynamic buttons” on page 4-6.

- Do you want users to be able to modify the database?

Although many menu commands are unavailable in a runtime application, users can still access the menu commands by opening your runtime database solution files in FileMaker Pro. (For a complete list of available menu commands, see appendix A, “Feature comparison of the runtime application and FileMaker Pro.”)

If you do not want users to modify your files, you can create passwords to prevent them from opening the files in FileMaker Pro. (See “Protecting your database solution files” on page 4-7.)

You can also make your files permanently unmodifiable by selecting the **Permanently prevent modification of database structure** option in the Developer Tool. (See “Removing design access to your databases” on page 6-11.)

- Will this be a cross-platform runtime database solution?

For advice on handling fonts, graphics, and general cross-platform issues, see “Design tips for cross-platform solutions” on page 4-11.

- How will you provide updates for your users?

Plan ahead for the time you may want to update your runtime database solution files. You can make the process easier for users by providing scripts in your primary file to export their data and import it into the updated solution. See “Importing data into an upgraded runtime database solution” on page 4-17 for an overview.

- Will your users be printing reports or other information from your runtime database solution?

It’s a good idea to set document margins in FileMaker Pro if your runtime database solution will be printed from a variety of printers. See “Specifying page margins” in the *FileMaker Pro User’s Guide* or see FileMaker Pro Help.

- Do you want users to be able to perform spell checking on records?

The FileMaker Developer Tool does not automatically include a spelling dictionary in the runtime solution. You must choose the main dictionary for your solution database from within FileMaker Pro. Once a main dictionary is selected for the solution database, your users can add or modify a user-defined dictionary to the runtime application. See the FileMaker Pro onscreen Help for details about main and user dictionaries.

Considerations for Kiosk mode

When you create a solution to run in Kiosk mode, you need to consider if you want to make your solution a stand-alone application, how the user will navigate your solution, and how the user will be able to quit the solution.

Displaying a database in Kiosk mode

To display a solution in Kiosk mode, you must either create a stand-alone application or assign a limited access password to the primary file. See “Protecting your database solution files” on page 4-7 for information about assigning a limited password, and “Creating Kiosk-mode solutions” on page 6-9 for information about creating a Kiosk-mode solution.

Navigating in Kiosk mode

The primary file is the main database that users see first in your Kiosk solution. Because Kiosk mode does not contain any menus or window controls, the primary file must contain buttons that users can click to navigate through the solution, close the files, and to quit FileMaker Pro or the runtime application.

To decide how users will navigate your Kiosk database solution, start by planning your navigation design on paper. Decide what will happen when each button is clicked, and give users a way to get back to the beginning of your system from each layout. To further control what users see, create startup scripts that display a specific layout when a file is opened.

- If your Kiosk solution will be run with a touch screen, use large buttons and allow space between buttons.
- Try to limit the number of options available on one screen.
- Because Preview mode disables buttons, make sure that any script step to enter Preview mode is followed by a Pause/Resume script step and specify an amount of time the script should remain in Preview mode; then make the script return to Browse mode.

Note When a Kiosk database solution is open, access to the operating system is limited. On Windows machines, you can press Alt+Tab to go to another application from your Kiosk database solution; on Mac OS machines, you can press \mathcal{H} +Tab.

Closing a Kiosk mode solution

If there is no Quit or Exit button available in your Kiosk solution, users must force-quit the application by pressing Alt+F4 (Windows) or by pressing Option + \mathcal{H} + Esc (Mac OS). Force quitting is not recommended because it can cause data corruption or damage open files.

To ensure that users can always access the primary file and quit the application, you can:

- provide a startup script in each auxiliary file that opens the primary file
- place a “Main Menu” button in each auxiliary file that returns to the primary file
- include a Quit button in the primary file

See “Emulating menu commands and window controls” on page 4-6 for information about creating buttons and scripts that emulate missing menu options and window controls.

Example Kiosk solution

FileMaker Developer provides a Kiosk Solution Example on the *FileMaker Developer* CD:

`\Developer Extras\FileMaker, Inc\Examples\Kiosk Solution\`

The Kiosk example solution has a blank, limited password assigned. The master password is kiosk. To run the example solution in Kiosk mode, leave the password blank. To open the example database in FileMaker Pro with full menu bar access, type kiosk in the password box.



The World Class Burger Company's database solution in Kiosk mode



The Kiosk Solution Example contains buttons for navigating between screens, adding and deleting records, and quitting the database

Using scripts to control your solution

You can use scripts to automate much of your database solution, control startup behavior, emulate menu commands and window controls, navigate, and much more.

Note The Developer edition of FileMaker Pro includes a Debug Scripts command to help you create and test scripts.

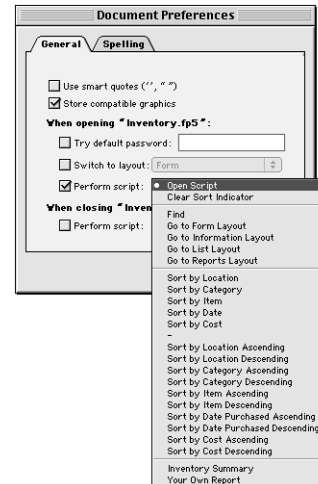
For more information about scripting in FileMaker Pro, see “Creating scripts to automate tasks,” in the *FileMaker Pro User’s Guide* or see FileMaker Pro Help. For information about using the Debug Scripts command, see FileMaker Pro Help.

Creating startup scripts

Startup scripts are useful for controlling what appears when users open a file in your solution. For example, you can control which layout is displayed and the size, position, and magnification of the window. See “Creating a custom web site using a database layout” on page 8-7 for information about using a startup script with Instant Web Publishing.

To create a startup script:

1. In FileMaker Pro, open the file that will contain the script, such as the primary file.
 2. Choose Scripts menu > ScriptMaker and create a script to perform the desired actions.
 3. Choose Edit menu > Preferences > Document.
- Mac OS X: choose FileMaker Developer application menu > Preferences > Document.
4. In the When opening “document.fp5” area, select Perform script and choose the script you created from the pop-up menu.



5. Click OK.

Now, the script that you specified is automatically performed whenever the file is opened.

Centering a database window in the Kiosk screen

To center your databases in the middle of a Kiosk screen, create a startup script that uses the Toggle Window and Set Zoom Level script steps. When a file is opened in Kiosk mode, two things will happen:

- The database window snaps to fit layout objects at the right and bottom edges of the layout.
- If the window is smaller than the available screen area, it is centered in the middle of the screen.

Important Before using the Toggle Window script step, perform any script steps that affect the window display area (such as Go to Layout or Toggle Status Area). Once the window area is determined, add the Toggle Window script step.

The Toggle Window script step may cover up a window that has a Quit Application button. Be sure that users can close the Kiosk database solution easily.

Emulating menu commands and window controls

Use the following script steps to emulate menu commands and window controls.

To emulate these interface elements	Create buttons with these attached script steps
Menu commands	Script steps for any menu command (for example, Sort, New Record, and Quit Application)
Zoom controls	Toggle Window or Set Zoom Level
Status area control	Toggle Status Area
Mode pop-up menu	Enter Browse Mode, Enter Find Mode, or Enter Preview Mode
Vertical scroll bar	Scroll Window (if the layout is longer than one screen) or View as List (if browsing)

To emulate these interface elements

Horizontal scroll bar

Window size and position

Create buttons with these attached script steps

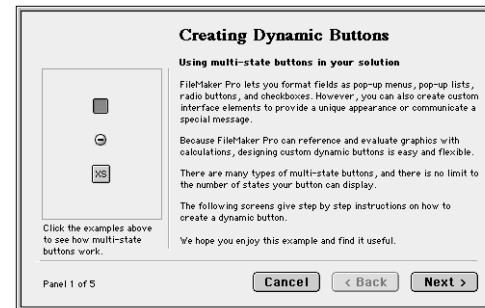
Scroll Window (if the layout objects are wider than one screen)

Open File and Toggle Window in a startup script, or set desired size and position before creating the Kiosk solution

Creating dynamic buttons

FileMaker Developer provides a database named Creating Dynamic Buttons.fp5—a short tutorial for how to make a button perform multiple scripts and appear in multiple states of operation (such as depressed when clicked, raised when not clicked).

\\Developer Extras\FileMaker, Inc\Examples\Creating Dynamic Buttons\Creating Dynamic Buttons.fp5



This tutorial shows how to make a button appear to dynamically change when it's clicked

Tip In Layout mode, you can see the boundaries of objects that have scripts attached to them by choosing View menu > Show > Buttons.

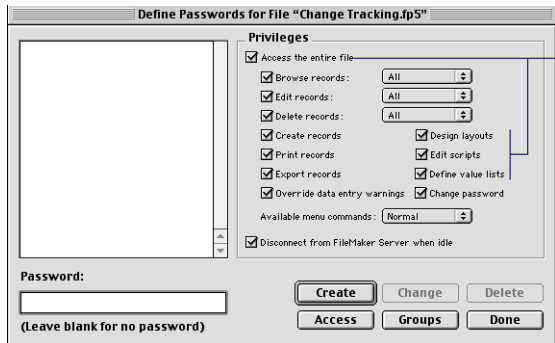
For more information, see “About working with objects on a layout” and “Using buttons with scripts” in the *FileMaker Pro User's Guide* or see FileMaker Pro Help.

Protecting your database solution files

If passwords have not been defined, your solution files can be opened in FileMaker Pro, and users can access and modify scripts, field and relationship definitions, access privileges, and layouts. You may want to set a master password that gives you access to the entire file and another password that gives users limited access.

To create a master password and a limited password for users:

1. In FileMaker Pro, choose File menu Access Privileges > Passwords.
2. In the Password text box, type a password to use for the master password.
3. Select all the privileges you want for the master password, then click Create.
4. In the Password text box, type a password to give users limited access.
5. Remove access privileges for Access the entire file, Design layouts, Edit scripts, Define value lists, and Change password.



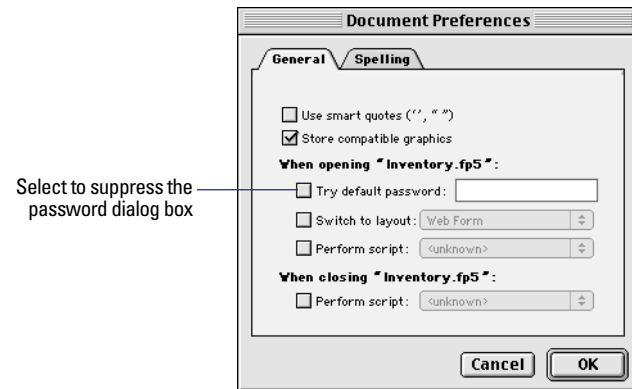
Leave these privileges deselected for limited access

6. Click Create.

You can set an option to automatically enter the limited access password and suppress the password dialog box from appearing when the file is opened.

To automatically open the solution with the limited access password:

1. In FileMaker Pro, choose Edit menu > Preferences > Document.
Mac OS X: choose FileMaker Developer application menu > Preferences > Document.
2. Select Try default password and type the limited access password in the box.
3. Click OK.



Select to suppress the password dialog box

Tip Press and hold the Shift key (Windows) or Option key (Mac OS) while the file is opening to force the password dialog to open. You can then enter your full-access or master password and modify the file.

You can permanently remove access to the Access Privileges menu command after you've defined the passwords, as well as access to other menu commands, by selecting the Permanently prevent modification of database structure option in the Developer Tool. See "Removing design access to your databases" on page 6-11 for information.

Providing user documentation

There are several ways that you can provide documentation for your database solution, including printed manuals, an online Help system, and an About layout screen that is available from any layout in the solution.

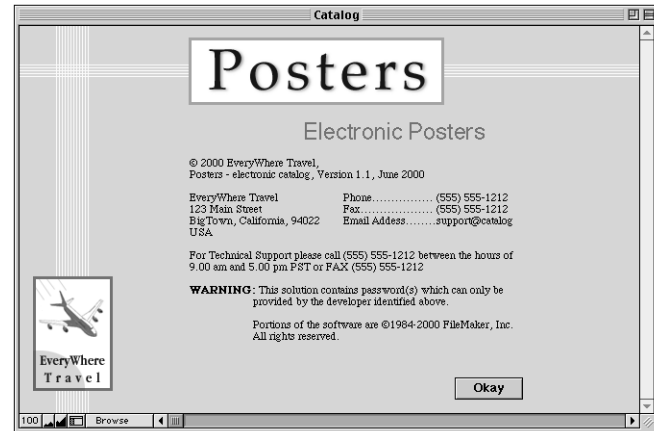
Create custom About and Help screens that document what your database solution is, how to use it, and where users can go for more information. Then use the Developer Tool to attach scripts to menu commands that open the About and Help screens. These menu commands are made available in FileMaker Pro along with menu commands for FileMaker Pro Help and About FileMaker Pro. For runtime database solutions, these menu commands will replace the FileMaker Pro Help and About FileMaker Pro menu commands in the runtime applications.

Note The FileMaker Pro Help system is not available in FileMaker Pro runtime applications. However, Status Bar Help (Windows) Balloon Help (Mac OS), and Help tags (Mac OS X) are available. In addition, you can provide What's This? Help for runtime database solutions that run on Windows machines. See "Providing What's This? Help (Windows)" on page 4-10 for information.

Creating an About layout

For runtime database solutions, the FileMaker Developer license specifies that you must create an About layout (or screen) that provides information for your users on how to contact you for technical support.

An About layout helps protect your database solutions in the event that your users approach FileMaker, Inc. for passwords. FileMaker, Inc. uses the About layout to distinguish databases created by developers using FileMaker Developer rather than users of the retail version of FileMaker Pro.



Example of an About layout

For more information about what is required to appear in the About layout for runtime database solutions, see "Your responsibilities as a developer" on page 4-15.

For a demonstration of an About layout in the primary file of a runtime database solution, see the runtime solution example on the *FileMaker Developer CD*:

`\Developer Extras\FileMaker, Inc\Examples\Runtime Solution\`

To create an About layout:

1. Create a blank layout in the primary file of your database solution and include the word "About" in the layout name. (In FileMaker Pro, choose **View** menu > **Layout Mode**, choose **Layouts** menu > **New Layout/Report**, type `About <your solution>` in the **Layout Name** box, select **Blank Layout** and then click **Finish** to create the layout.)

Note For runtime database solutions, you must include the word "About" in the layout name. You must also include certain specific information in the layout. See "Your responsibilities as a developer" on page 4-15 for details.

2. Include your logo, any other graphics, and your company information.

3. Notify users if the solution files are protected with passwords or if database design access has been removed.

See “Your responsibilities as a developer” on page 4-15 for the exact legal wording.

4. Create a button that lets your users return to the main layout of the primary file. (To create the button, use the button tool or a button image with a script attached to it.)

See “Creating dynamic buttons” on page 4-6 and “Using buttons with scripts” in the *FileMaker Pro User’s Guide* or see FileMaker Pro Help for information.

5. Choose Scripts menu > ScriptMaker and create a script that goes to the About layout. Include the word “About” in the script’s name.

About <solution name>

6. Use the Developer Tool to create a menu command that displays the About layout.

The Developer Tool uses the script’s name for the name of the menu command. See “Adding a custom script to an About menu command” on page 6-11.

You can also display the About layout by:

- Creating a script that displays the layout each time the user opens a solution file. (See “Creating scripts to automate tasks” in the *FileMaker Pro User’s Guide* or see FileMaker Pro Help for information.)
- Setting the document preferences for the primary file to always display the About layout when the runtime application is started:

1. Choose Edit menu > Preferences > Document

Mac OS X: choose FileMaker Developer application menu > Preferences > Document.

2. Select Switch to layout, and choose the About layout from the pop-up menu.

Creating a custom Help layout or file

Create a Help layout or file that provides instructions for how to use your custom solution and enter data. Then, create a script in the primary file of your solution to display the Help system, and use the Developer Tool to make the script available as a command in the Help menu. (See “Adding a custom Help script command to the Help menu” on page 6-13 for information.)

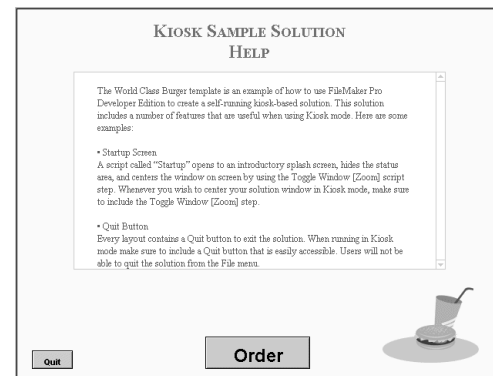
FileMaker Developer provides three examples of integrated Help systems that you can study and get ideas from. Open the following example files on the *FileMaker Developer* CD:

\\Developer Extras\FileMaker, Inc\Examples\Kiosk Solution\Menu.fp5

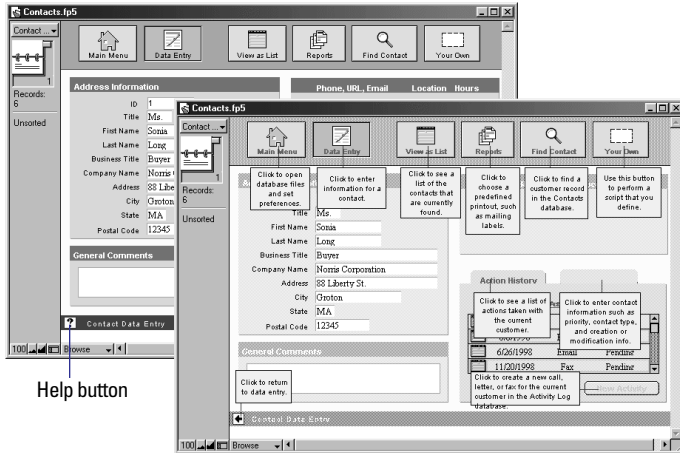
\\Developer Extras\FileMaker, Inc\Examples\Runtime Solution\Cat_help.usr

\\Developer Extras\FileMaker, Inc\Examples\Relational\Main Menu.fp5

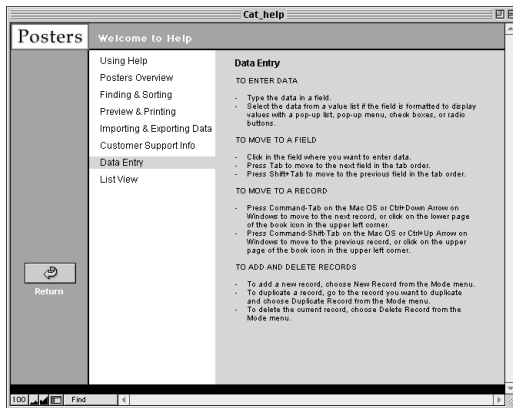
Each solution provides a button for users to close the Help window and return to the main layout.



The Kiosk Solution Example contains a single Help layout



In the Relational Example, a custom Help layout appears when you click the ? button in each auxiliary file

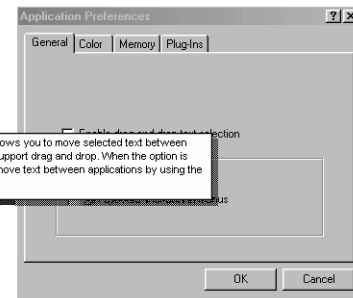


The Runtime Solution Example contains a separate Help file, available by clicking a Help button or choosing Posters Help from the Help menu

In addition to an online Help system, you should include printed documentation that explains how to install your database solution and briefly how to use it in case users are not able to open your solution files. See the paragraph below, “Including printed documentation,” and “Documenting the installation procedures” on page 7-5, for information.

Providing What's This? Help (Windows)

When you create FileMaker Pro runtime database solutions for Windows users, the Developer Tool generates a What's This? Help file that you can distribute with your databases. When users select an item in an open dialog box and click the ? button or press Shift+F1, a separate What's This? Help window opens describing the option in the dialog box.



Select an option in the dialog box and press Shift+F1 to display What's this? Help for that option

See chapter 7, “Distributing FileMaker Pro runtime database solutions” for more information.

Including printed documentation

Here are some suggestions for items to include in the printed documentation:

- How to install your bundled solution
- How to upgrade to new solution files

- How to use your Help system
- How to use What’s This? Help (Windows)
- How to start the database solution (see “Starting your runtime database solution” on page 7-5)
- What to do in case of a damaged file (see “Recovering damaged files” on page 7-6)
- How to reach you for technical support

You might also want to include the following recommendations:

- Tell your users not to rename any solution files (except the runtime application), or they may be unable to run your solution.
- Recommend that users back up their data regularly. You might want to automate some of the process by including scripts that save copies of the solution files.

For more information on automating a backup procedure, see the FileMaker Pro Help.

Important In the event that your runtime database solution files become damaged, make sure users have access to your technical support e-mail address or telephone number in your printed documentation or in a text file. If a database file is damaged, they may not be able to access the About layout in your solution to find out how to contact you.

Design tips for cross-platform solutions

If you’re developing a database solution to be run on both Windows and Mac OS machines, there are many issues that require planning and consideration—for example, which font technology, colors, and design conventions to use.

Creating a consistent appearance

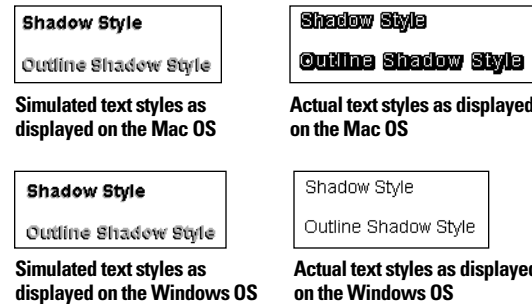
To ensure a consistent look for files across platforms, try to use the same font technology, for example TrueType. It is best to use fonts with identical names, styles, and metrics on both platforms. When fonts used to view a file are different from the ones used to create the file, there can be problems with word wrapping and placement.

Most font vendors supply Windows and Mac OS versions of the same font. Contact your font vendor for more information.

When you move a file from one platform to another, if the font used to create the file isn’t available, another font will be substituted. In FileMaker Pro for Windows, you can specify how fonts are substituted in the Microsoft Windows Registry Editor. You can add more fonts to the Registry Editor to include more font substitutions. If there is no match and no substitute is defined, FileMaker Pro changes the font to Arial. In the Mac OS, FileMaker Pro attempts to substitute a similar font.

Simulating outline and shadow text styles

The Windows platform doesn’t offer outline and shadow text styles. If you want a shadow effect for titles or headings, simulate the look by positioning two copies of a text block so that the foremost copy appears to cast a shadow against your layout.



To simulate a shadow effect for text:

1. Add text to your layout.
2. Set its text color to a shade of gray. (Choose Format menu > Text Color and choose gray from the color palette.)
3. Duplicate the text object. (Select the text object and choose Edit menu > Duplicate.)
4. Set the text color of the duplicate to black.
5. Move the duplicate back over the original.
6. Press the up-arrow key once, then the right-arrow key once.

You can vary the offset to achieve whatever shadow effect you prefer.

7. Check on both platforms to be sure the shadow effect appears as you intended.

To simulate a combined outline and shadow effect, reverse the gray and black text colors in steps 2 and 4.

Using common character sets

When sharing files between the Windows and Mac OS platforms, use characters that exist in both character sets when available. While international characters are usually available, many of the upper ASCII characters don't match across platforms. Keep in mind that special characters are handled differently across fonts.

The following are examples of characters that won't display properly across platforms.

Windows-only characters:

- $\frac{1}{4}$ symbol
- $\frac{1}{2}$ symbol
- $\frac{3}{4}$ symbol

Mac OS-only characters:

- \neq (not equal)
- \geq (greater than or equal to)
- \leq (less than or equal to)
- $\sqrt{\quad}$ (square root/check mark)

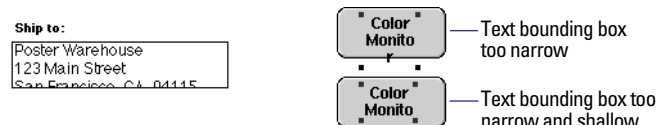
See the topic "Designing cross-platform databases" in the onscreen Help for more information.

Designing text layouts for cross-platform solutions

Keep the following tips in mind when designing a text layout that will display in both Windows and the Mac OS.

- Leave plenty of room around a text block.

Even when FileMaker Pro is able to match fonts, there can be subtle differences in character width and line spacing when fonts are substituted on another platform. FileMaker Pro supports fixed text object sizes, so that you can resize a text object to be longer than the text within it. This allows you to anticipate changes in font size. Resize your text objects so they are a little wider than the default size to prevent text from wrapping to a second line when a different font is substituted.



Make sure fields and text objects are large enough to accommodate substituted fonts

- Check the alignment of the field labels and their associated fields—they should both be aligned in the same direction.

If you put a left-aligned label over a column of right-aligned numbers, for example, the report might look fine on your computer. But font substitution could cause field labels to shift when the file is opened on another computer. If a wider font is substituted on the second computer, your column heading will shift to the right. If a more narrow font is used, the text will display too far to the left.

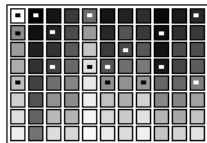
- Avoid mixing text and fields because character spacing may vary. If necessary, use merge fields—for example, to place a field in the middle of a sentence.

Using a common color palette

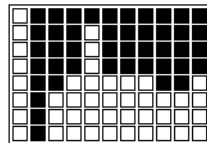
When 256 or more colors are available, FileMaker Pro will offer an 88-color palette that is virtually identical across platforms. (Close substitutes are used for 13 colors that do not match exactly.)

Windows computers using a standard VGA driver will only display 16 colors. Some older Mac OS computers may also be limited to 16 colors. The 16-color palette in FileMaker Pro varies slightly between Windows and the Mac OS, depending on the Windows color scheme you are using.

If you are building your files on a 256-color computer, you might want to use colors that will map well to 16-color systems. It also helps to know which colors map to black and which map to white for monochrome displays.



The dotted colors are available on 16-color systems



This palette indicates which colors map to black or white

Using graphics in cross-platform solutions

If you're storing and displaying graphics across platforms, be sure to select the document preference for **Store compatible graphics** before importing each graphic into your database file:

Choose **Edit menu > Preferences > Document**.

Mac OS X: choose **FileMaker Developer application menu > Preferences > Document**.

Two copies of the graphic image will be stored: the original version (for example, bitmap, metafile, or GIF) and one in PICT file format.

Graphics with gradients that are imported and stored as bitmaps will redraw faster on the screen than graphics imported and stored as PICT images. Additionally, PICT images containing gradients may have some quality degradation when displayed on Windows machines.

If your graphic image has an undesirable white border surrounding an irregularly shaped graphic, you should create a mask for the bitmap image. Refer to the documentation that came with your graphics program for more information.

Using QuickTime movies in cross-platform solutions

To use a Mac OS QuickTime movie with FileMaker Pro for Windows, first save the movie in a format playable on non Apple computers using a QuickTime editing application. For more information, visit the Apple web site at www.apple.com.

To save a QuickTime movie in a cross-platform format:

1. On your Mac OS machine, start your QuickTime editing program.
2. Open the QuickTime movie you want to convert.
3. Choose **File menu > Save As** and save the file with a new name and the .mov filename extension.
4. Select **Make movie self-contained**.

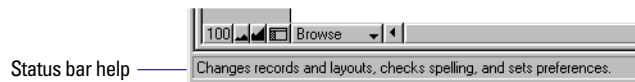
5. Select Make playable on non Apple computers.

The first time you play a movie from another platform, FileMaker Pro prompts you for the location of the movie. Keep the files in a common folder and avoid choosing the wrong file—this could lead to unexpected results.

Showing the status bar in Windows

FileMaker Pro for Windows includes an option to show the status bar, which allows users to specify whether status bar information at the bottom of a window is visible or not.

If your layouts are designed to take up most of the available screen space, users can deselect this option in FileMaker Pro or in your runtime application to make more room.



To show the status bar:

- Choose View menu > Status Bar.

A check mark next to the menu item indicates that the option is selected.

Using separate scripts for printing

Scripts that include Page Setup/Print Setup and Print commands are not 100% compatible across platforms. The print steps in the ScriptMaker feature rely upon the current printer driver in order to determine the paper sizes that are available, page orientation, and so on. Even if a computer using Windows and another using the Mac OS are connected to the same printer, the drivers themselves are significantly different—so FileMaker Pro is unable to restore page setup and print options across platforms.

To work around this, you can do the following:

1. Create separate scripts for Windows and Mac OS. First, open the file on your Windows machine and create the script for printing from Windows. Then, move the database file to the Mac OS machine and create the script for printing from the Mac OS.
2. In ScriptMaker, use the If script step and the Status (CurrentPlatform) function in both scripts to determine whether to run the Windows or the Mac OS script. For more information, see “Using the Status (CurrentPlatform) function” next.
3. Make sure the Perform Without Dialog option for the Print script step is not selected. This will allow users to change the setup options before they print.
4. In Layout mode, choose Layouts menu > Layout Setup, select Fixed Page Margins, and specify margins for Top, Bottom, Left, and Right.

Using the Status (CurrentPlatform) function

FileMaker Pro includes a status function that lets you determine the platform on which the database solution is being run. This allows you to perform different script actions such as changing to a different layout or performing a platform-specific script step. The Status (CurrentPlatform) function returns an integer value that identifies the current operating system. For example, the function returns 1 in Mac OS 9 or earlier, and -1 in Mac OS X. See the onscreen Help for complete information about integer values.

Use this function with the If script step to perform different actions depending on the current platform.

```
If ["Status (CurrentPlatform) = 1]
    Perform Script [Sub-scripts, "Print in Mac OS"]
Else
    Perform Script [Sub-scripts, "Print in Windows"]
End If
```


Creating platform-specific scripts

Although most ScriptMaker steps work on both platforms, some script steps rely upon platform-specific features. The following steps are platform-specific:

Windows-only script steps

- DDE Execute
- Send Message
- Insert Object
- Update Link

Mac OS-only script steps

- Send Apple Event
- Perform AppleScript
- Speak

The Windows Print Setup script step is translated to the Mac OS Page Setup script step, and vice versa. Note that print options depend on the printer driver, so that options you save with a script on one machine may not be available on another. The Send Message script step (Windows) is translated to the Send Apple Event script step (Mac OS), and vice versa.

Other platform-specific script steps are skipped when run on a different platform. Also, platform-specific script steps are shown in italics when viewed on a different platform.

Your responsibilities as a developer

FileMaker, Inc. has a policy of established procedures for decoding and repairing passwords. If a customer complies with these procedures, then FileMaker, Inc. may decode passwords, and/or supply data or a repaired file for the customer.

If you distribute FileMaker Pro runtime files with passwords, or you have removed master level design access and you do not want FileMaker, Inc. to decode passwords, repair access privileges, and/or return data to a customer who requests such services, you must do the following:

1. Notify your customers in writing and keep a record of such notice that your database solution contains passwords and/or data that can only be provided by you.
2. Every file in your runtime database solution must contain an About layout accessible from any layout in the database. See “Adding a custom script to an About menu command” on page 6-11 and “Creating an About layout” on page 4-8.
3. The layout name must begin with the word “About.”
4. The About layout must contain these items:
 - Solution Name
 - Your company information: company name, address, city, state, postal code, country, and phone number
 - Your support policy (for example, how and when you are available for technical support)
5. If your databases contain passwords, and you do not want FileMaker, Inc. to decode and repair passwords and access privileges for your database, the About layout must contain this exact warning:

“USER WARNING: This database solution contains password(s) that can only be provided by the Developer identified above.”
6. If design access has been removed from your database solution by selecting the Permanently prevent modification of database structure option in the Developer Tool, then the About layout must contain this exact warning:

“USER WARNING: This file is not customizable. Contact the above named Developer for information on customizing this database solution.”

The password protection in a FileMaker file should not be viewed as an absolute barrier that will prevent a customer from accessing files. FileMaker, Inc. cannot guarantee that a customer will not be able to identify or bypass the password without any assistance from FileMaker, Inc. Therefore, FileMaker Inc. recommends that you take appropriate steps to protect your consulting and development efforts without relying solely upon the password.

If you have a dispute with your customer, you must resolve this dispute directly with the customer. FileMaker, Inc. is unable to, and will not, attempt to resolve such disputes.

Testing before and after creating your solution

You should verify the functionality of your database solution by testing it thoroughly before and after you customize it with the Developer Tool.

Here are some suggestions for ensuring the quality of your custom database solution:

- Verify every function and option in your solution. If you're developing a cross-platform solution, test it on both Windows and Mac OS platforms. (See "Design tips for cross-platform solutions" on page 4-11 for information.)
- Make sure your runtime database solution does not use a standard FileMaker Pro feature that is hidden or disabled in the runtime application. See appendix A, "Feature comparison of the runtime application and FileMaker Pro" for information.
- Verify that all scripts and buttons work as expected. This is especially important if you are displaying your solution in Kiosk mode.
- Verify your installation procedures and test other instructions in the documentation.

- Verify that your database layouts display well on monitors with different color capabilities and resolutions and on the smallest size monitor your users may be using.
- Test your runtime database solution with actual data. This is especially important if users are upgrading from earlier versions of the runtime application and need to import data into new solution files.
- Make sure all the auxiliary files and DLLs (Windows) are present.
- Show your database solution to intended users to uncover any usability issues.
- Install your bundled database files on a completely different computer to verify that all the files associated with the primary file can be found.
- If you're assigning passwords or permanently removing design access, test all access levels. Make sure your database solution contains an About layout that notifies users of the level of access you're providing. See "Creating an About layout" on page 4-8 and "Your responsibilities as a developer" on page 4-15 for information.

Important You should keep an unbound version of any runtime database solution files, especially if you've permanently removed design access. (See "Removing design access to your databases" on page 6-11.)

Converting and upgrading solution files

If you have developed a FileMaker Pro runtime database solution using the Solutions Development Kit (SDK) for FileMaker Pro 3.0 or earlier or the Binder utility in the FileMaker Pro 4.0 Developer Edition, you may wish to upgrade your solution and provide your users with the converted files.

In general, to upgrade your solution, you need to do the following:

1. Open the original solution database files in the current version of FileMaker Pro.

The original solution files will be renamed “<Filename> Old” and the converted files will keep the original name. If the solution files use relationships or external scripts, the converted files will continue to work together.

2. If you want, update your databases to take advantage of newer FileMaker Pro features, such as the enhanced value lists.

3. If necessary, create scripts to import users’ existing data from the old runtime database solution into the new, upgraded solution.

See “Importing data into an upgraded runtime database solution” next.

4. Use the Developer Tool to bind the solution files into a new, upgraded runtime database solution.

See “Binding your databases into a runtime database solution” on page 6-6.

Note There may be a conflict with the icons for the runtime application and database files if your users have an earlier version of your runtime solution on their machines. (Your users will see the old FileMaker 4.0 Developer Edition icons.) To avoid this, use a different filename extension.

5. Distribute the new upgraded runtime database solution and provide instructions for how users can upgrade their files by opening the old files in the new runtime application and importing their data.

Importing data into an upgraded runtime database solution

You can include scripts in the new runtime database solution files that allow users to import records from the old runtime files.

To prepare your upgraded solution for importing data:

1. Open the original solution database files in the current version of FileMaker Pro.

The converted filenames should be the same as the original filenames.

2. Add new features as desired to the converted solution files.

3. Place the original files in a folder named “Old Solution Files.”

4. In each of the new converted files, create an Import script that imports all of the records from the old corresponding solution file.

5. Also, in each of the converted files, create an Open File script that lets users open the old solution file from the new runtime application.

Users can start the new runtime application, open an old solution file, and perform find requests to locate a subset of the records. Then they can execute the Import script to import only those records.

Tip If you have scripts in the original solution files that help your users reduce the found set (for example, a script that enters Find mode and pauses), you can call that script before the Import script step.

6. Use the Developer Tool to bind your converted files into the new runtime database solution.

7. Test your scripts carefully.

It’s a good idea to use sample data to make sure the records are importing properly and data is going to the correct fields. Test the data on another machine to make sure the external scripts work.

8. Distribute the new solution files that contain the Import scripts.

9. Provide instructions telling users how to import data into the new solution files and set appropriate time expectations. If their solution files are large, the conversion and import process may be lengthy.

Use file size information to calculate how much disk space is necessary for the conversion to go smoothly. Keep in mind that when users open an old solution file from the new runtime application, the file is copied before it’s converted. So, if a solution file is 2 MB, users will need at least 4 MB available to convert it.

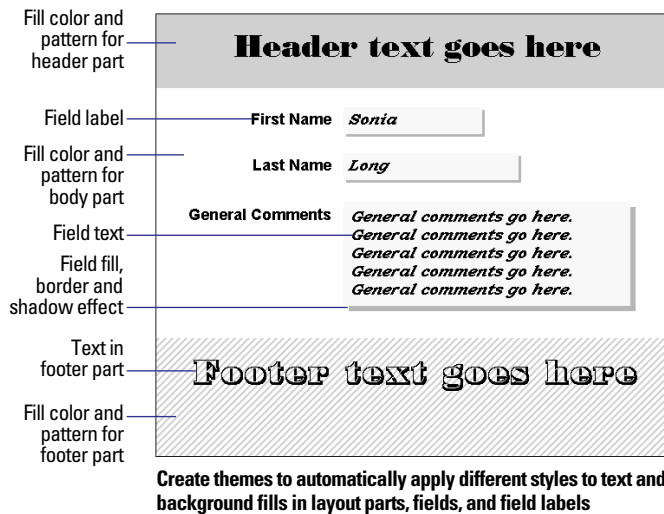
When users execute the Import script, the runtime application finds the original solution file, converts it, then imports the records into the new solution file. During the script, the runtime application temporarily converts a copy of the old file to the FileMaker Pro runtime version, and deletes it when the original file is closed. Your users should delete the original solution files after making a backup copy of the new solution files.

Chapter 5

Creating custom layout themes

FileMaker Pro uses a variety of layout themes to describe the colors, patterns, fonts, and borders of text, fields, and parts in a new layout.

A FileMaker Pro theme is defined in an Extensible Markup Language (XML) document that can be read and edited in text editors (such as Wordpad for Windows or BBEdit for Mac OS) or XML editors (such as XML SPY or XML Pad). You can customize an existing theme or create your own, and then use the New Layout/Report assistant to apply the custom theme when you create layouts for your databases. You can modify attributes defined by the theme in Layout mode after the layout is created. However, you can't apply a theme to an existing layout.



Note A FileMaker Pro theme is not a stylesheet and does not contain positioning information for objects on a layout.

For information about:

- using layout themes and designing layouts, see the *FileMaker Pro User's Guide*
- XML and its uses, see the product support pages for FileMaker Developer on the FileMaker, Inc. web site at www.filemaker.com
- publishing your database on the Web in XML format, see chapter 10, "Using FileMaker Pro XML to deliver your data on the Web"
- creating a custom web site using a database layout, see page 8-7

Modifying a FileMaker Pro theme

The Developer edition of FileMaker Pro includes theme files, which you can modify. A FileMaker Pro theme file can contain more than one theme—for example, the `Blue_gold.fth` file contains two themes: "Blue and Gold Screen" (for viewing onscreen) and "Blue and Gold Print" (for printing).

Important The XML for a layout theme must be well-formed and comply with the required syntax. See "Basic requirements for a theme file" on page 5-3 and "Checking your theme document for errors" on page 5-11.

To modify a theme:

1. Make a duplicate copy of the theme file (for example, `Blue_gold.fth`) in the Themes folder.

FileMaker Developer 6\Themes\

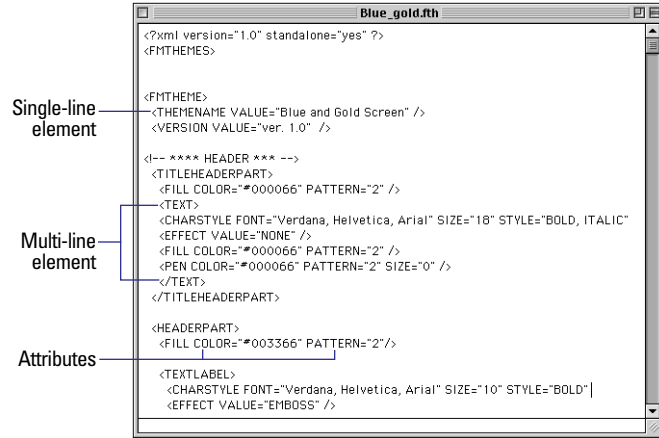
or

FileMaker Developer 6\FileMaker Pro.app\Contents\MacOS\Themes\

2. Rename the copy and include the .fth extension with the new filename. Keep the new file in the Themes folder.

In order for the New Layout/Report assistant in FileMaker Pro to display a theme option, the theme file must reside in the Themes folder and it must have the .fth filename extension.

3. Open the theme file in your text editor.



4. Change the name of a theme by replacing the value of the THEMENAME element with a new name.

```
<THEMENAME VALUE="Purple and White Screen" />
```

Note If your THEMENAME value contains any upper-ASCII characters, use the HINT attribute to ensure that the theme name will appear on both the Windows and Mac OS platforms. See “Valid values for theme attributes” on page 5-7 for information.

5. Change the values of other elements and attributes as desired.

For example, to change the background fill color of the body part in a layout to a light purple, change the color hexadecimal (hex) value to #9933CC:

```
<BODYPART>
```

```
<FILL COLOR = "#9933CC" PATTERN = "2" />
```

See the table in “Valid values for theme attributes” on page 5-7 and “Finding values for patterns and colors” on page 5-9 for guidelines.

6. Remove any elements that you don’t want to specify. Be sure to remove the entire single-line or multi-line element including its start and end tags.

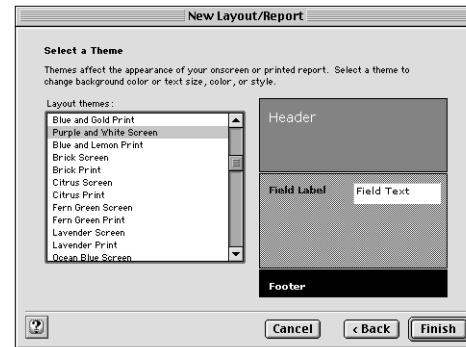
See “Removing elements from a theme file” on page 5-4 for information.

7. Scroll down to the next FMTHEME element and repeat these steps to change the THEMENAME value, and other elements as desired.

8. Save the file in text format with the filename extension .fth in the Themes folder inside the FileMaker Pro folder.

In FileMaker Pro, each new THEMENAME value will appear in the New Layout/Report assistant as a Layout Theme option.

9. In FileMaker Pro, choose Layouts menu > New Layout/Report to use your theme.



Names of your custom themes appear as options in the New Layout/ Report assistant

If your new themes don’t appear in the New Layout/Report assistant, you might have made a syntax error. See “Checking your theme document for errors” on page 5-11 for information.

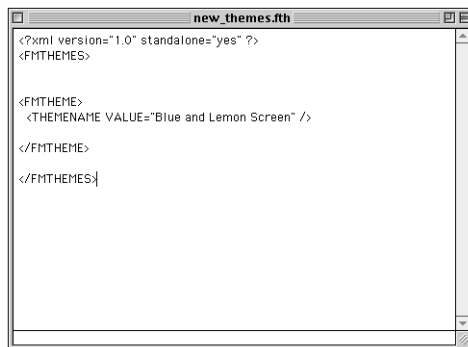
Basic requirements for a theme file

FileMaker Pro layout themes are described in an XML document saved in text file format. Each text file must have the .fth filename extension and reside in the Themes folder inside the FileMaker Developer application folder so FileMaker Pro can display the themes in the New Layout/Report assistant.

XML resembles HTML in many ways—however, unlike HTML the XML for layout themes must be well-formed and comply with the required syntax. Omitting a required element or attribute, or mismatching start and end tags will result in an unusable document and FileMaker Pro will be unable to parse the XML or display the themes in the New Layout/Report assistant.

Minimum required XML elements for themes

Every theme file must begin with an XML-document processing instruction that declares it as an XML document using the XML 1.0 specification. In addition, an XML document for a FileMaker Pro layout theme must contain the <FMTHEMES> and </FMTHEMES> start and end tags for the file. This FMTHEMES root element can contain one or more FMTHEME element.



Minimum elements required for a theme file

Containing all of your themes (FMTHEME elements) in one file is useful if you want to organize the way that themes appear in FileMaker Pro. The order that FMTHEME elements are listed in the file determines the order that the THEMENAME values will appear in the New Layout/Report assistant.

Note Values for the THEMENAME element can contain any characters from the ASCII character set. However, if you’re using an XML editor to write your themes or if you plan to use the themes on different platforms, certain measures must be taken.

XML editors expect these characters to be coded as character entities:

Character	Coded as
ampersand (&)	&
less than (<)	<
greater than (>)	>
apostrophe (')	'
quote (")	"

Using the character instead of the character entity results in an error from the XML editor. However, FileMaker Pro does not reinterpret character entities—values in the THEMENAME element will appear exactly as typed. You can avoid the problem by using a text editor to create your themes or simply ignore the error from the XML editor. Your theme names will appear as you write them in the New Layout/Report assistant.

If you’re planning to use your themes on Windows and Mac OS platforms, use the HINT attribute to ensure that upper-ASCII characters (such as the accent mark) appear correctly on both platforms. See “Valid values for theme attributes” on page 5-7 for more information.

Removing elements from a theme file

The FileMaker Pro layout theme files contain multi-line elements for fields, field labels, text, and every part in a layout. Each of these elements contains other multi-line elements and single-line elements. You can remove any of these elements, but you must remove the entire element—that is, everything inside the element's start and end tags and the start and end tags as well.

FileMaker Pro will use default values for any elements you remove (see “Specifying default values for themes” on page 5-9).

A single-line element, such as the PEN element, begins with <PEN and ends with /> on a single line, and looks like this:

```
<PEN COLOR="#000066" PATTERN="2" SIZE="0" />
```

A multi-line element has start and end tags that look like this:

```
<BORDER>
</BORDER>
```

To remove a multi-line element, delete the start and end tags and all elements contained within them. For example, to remove a multi-line BORDER element in the Blue_gold.fth file, delete all three lines:

```
<BORDER>
    <PEN COLOR="#000000" PATTERN="2" SIZE="1" />
</BORDER>
```

XML elements for layout parts

An FMTHEME element can contain any of the following multi-line elements to describe the parts in a FileMaker Pro layout. Each layout part element contains additional elements to describe the background fill, text, field labels, and fields in the layout part.

Elements for layout parts can be listed in any order within an FMTHEME element in the XML document. However, if two identical elements are listed (such as two BODYPART elements), FileMaker Pro will only use the attributes for the last one in the list.

This multi-lined element is used	To describe this layout part
<TITLEHEADERPART> </TITLEHEADERPART>	Title header — appears only once at the top of the first screen or page.
<HEADERPART> </HEADERPART>	Header — appears at the top of every screen or page (except the first one if there's a title header).
<LEADGRANDSUMPART> </LEADGRANDSUMPART>	Leading grand summary — appears at the beginning of a report and displays a summary field for all the records in a found set. (A layout can have only one leading grand summary part.)
<LEADSUBSUMPART> </LEADSUBSUMPART>	Leading subsummary — appears above the body part and displays a summary field for a subset of records as defined by the break field. You can describe up to nine leading subsummary layout parts — each LEADSUBSUMPART element must contain a PARTNUMBER element to distinguish it from the others.
<BODYPART> </BODYPART>	Body — appears in the middle of every screen or page. (A layout can have only one body part.)
<TRAILSUBSUMPART> </TRAILSUBSUMPART>	Trailing subsummary — appears below the body part and displays a summary field for a subset of records as defined by the break field. You can describe up to nine trailing subsummary layout parts — each TRAILSUBSUMPART element must contain a PARTNUMBER element to distinguish it from the others.

This multi-lined element is used	To describe this layout part
<TRAILGRANDSUMPART> <TRAILGRANDSUMPART>	Trailing grand summary — appears at the end of a report and displays a summary field for all the records in a found set. (A layout can have only one trailing grand summary part.)
<FOOTERPART> <FOOTERPART>	Footer — appears at the bottom of every screen or page (except the first one if there's a title footer).
<TITLEFOOTPART> <TITLEFOOTPART>	Title footer — appears only once at the bottom of the first screen or page.

Note Although a theme may include descriptions for every type of layout part, the type of layout you select in the New Layout/Report assistant determines which parts will appear in your new layout or report.

For information about layout parts, see “Creating and managing layouts and reports,” in the *FileMaker Pro User's Guide* or see FileMaker Pro Help.

XML elements for text

Any layout part element can contain FIELD, TEXTLABEL, and TEXT elements that are used to describe the characteristics of text or data in the part.

The FIELD element is used to describe text (data) in fields and field borders. The TEXTLABEL element is used for field label text. Field labels are displayed in the body part of a layout or in other parts such as the header of columnar reports. The TEXT element describes all other text that appears in a layout part (other than fields or field labels), such as title text in the header.

Description of XML elements and their theme attributes

The following tables describe the multi-line and single-line XML elements supported by FileMaker Pro in a layout theme document. (Unknown elements are ignored by FileMaker Pro.)

Table of multi-line elements

These multi-line elements	May contain these elements
FMTHEMES (required)	FMTHEME The FMTHEMES root element can contain multiple FMTHEME elements. (The order that they are listed determines how they'll appear in the New Layout/Report assistant.)
FMTHEME (required)	VERSION THEMENAME (required) THEMEDEFAULT Any or all layout part elements
TITLEHEADERPART	FILL
HEADERPART	FIELD
LEADGRANDSUMPART	TEXT
BODYPART	TEXTLABEL
TRAILGRANDSUMPART	
FOOTERPART	
TITLEFOOTPART	
LEADSUBSUMPART	FILL
TRAILSUBSUMPART	FIELD PARTNUMBER TEXT TEXTLABEL

These multi-line elements	May contain these elements
FIELD (text/data in a field)	BASELINE BORDER CHARSTYLE EFFECT FILL
TEXT (text in a part—except field labels or field data)	CHARSTYLE EFFECT FILL PEN
TEXTLABEL (text in a field label)	CHARSTYLE EFFECT FILL PEN
BASELINE (underlining field data)	ONOFF PEN
BORDER (field border)	PEN SIDES

BORDER and EFFECT elements share the same pen size—when used together, the pen size value that you set for BORDER will also apply to the EFFECT width. The pen size value must be greater than zero in order for an effect or a border to appear.

Table of single-line elements

The following table describes the correct syntax for all single-line elements and their attributes. Examples of attribute values are indicated in boldface. For a list of the possible values you can use for these attributes, see the next section, “Valid values for theme attributes.”

These single-line elements	Must contain these attributes
CHARSTYLE	COLOR FONT SIZE STYLE <i>Syntax example:</i> <CHARSTYLE FONT=“ Verdana, Helvetica, Arial ” SIZE=“ 18 ” STYLE=“ BOLD, ITALIC ” COLOR=“ #FFFFFF ” />
EFFECT	<EFFECT VALUE=“ EMBOSS ” />
FILL	COLOR PATTERN <i>Syntax example:</i> <FILL COLOR=“ #000066 ” PATTERN=“ 2 ” />
ONOFF	<ONOFF VALUE=“ OFF ” />
PARTNUMBER	<PARTNUMBER VALUE=“ 0 ” />
PEN	COLOR PATTERN SIZE <i>Syntax example:</i> <PEN COLOR=“ #000066 ” PATTERN=“ 2 ” SIZE=“ 0 ” />
SIDES	<SIDES VALUE=“ BOTTOM ” />

These single-line elements	Must contain these attributes
THEMENAME	<p>HINT (optional)</p> <p>VALUE</p> <p><i>Syntax examples:</i></p> <pre><THEMENAME VALUE="Fern Green Print" /></pre> <pre><THEMENAME HINT="MAC" VALUE="Grün Druck" /></pre>
THEMEDEFAULT	<p>VALUE</p> <p><i>Syntax example:</i></p> <pre><THEMEDEFAULT VALUE="CURRENT"/></pre> <p>For more information, see “Specifying default values for themes” on page 5-9.</p>
VERSION	<p>VALUE</p> <p><i>Syntax example:</i></p> <pre><VERSION VALUE="ver. 1.0"/></pre> <p>The VERSION element is currently not used by FileMaker Pro, but may be used in future versions.</p>

Valid values for theme attributes

The following table describes the attribute values supported by FileMaker Pro in a layout theme. Values must be enclosed within quotation marks (“ ”)—if a quotation mark is missing, FileMaker Pro is unable to parse the XML and cannot display the theme in the New Layout/Report assistant.

This attribute	Is used to describe these characteristics	And may contain these values
COLOR	<p>RGB color hex values for background fills, text, and borders in layout parts, fields, text blocks, and field labels.</p> <p>To display a color, the PATTERN attribute must not be set to “1” (which is transparent).</p> <p>FileMaker Pro themes use web-safe palette colors to ensure the color will appear the same on all computers.</p> <p>See “Finding values for patterns and colors” on page 5-9.</p>	<p>COLOR = “#FFFFFF”</p> <p>COLOR = “#33FF00”</p> <p>COLOR = “#CC9966”</p> <p>Or any 6-digit hex value (a combination of numbers 0-9 or letters A-F) preceded by the # symbol.</p>
EFFECT	<p>Embossing, engraving, or drop shadow 3-D effects for a field, text, or field label.</p> <p>When used in conjunction with a field border, the line width of the effect will be the same as the border pen size.</p> <p>The pen size value must be greater than zero in order for the effect or border to appear.</p>	<p>VALUE = “EMBOSS”</p> <p>VALUE = “ENGRAVE”</p> <p>VALUE = “DROPSHADOW”</p> <p>VALUE = “NONE”</p>

This attribute	Is used to describe these characteristics	And may contain these values
FONT	The name of the font. More than one font name can be specified, separated by commas. The first font available on a user's computer will be used in the layout. Note Font values are case sensitive and must be entered in sentence style with initial caps only.	FONT = "Times New Roman" FONT = "Geneva" FONT = "New York, Times, Helvetica, Arial" Or any other available font (In FileMaker Pro, choose Format menu > Font to see the available fonts.)
HINT	The name of the platform that the theme name is edited on. This attribute ensures that any upper-ASCII characters present in the THEMENAME value (for example, an accent over a letter in the theme's name) will appear in FileMaker Pro on both Windows and Mac OS.	HINT = "WIN" HINT = "MAC"
ONOFF	Whether a field's border should be displayed (turned on or off).	VALUE = "ON" VALUE = "OFF"
PARTNUMBER	To distinguish multiple leading or trailing subsummary parts in a layout. This attribute is ignored for all other parts. FileMaker Pro supports values 0 through 9 and ignores any other value.	VALUE = "0" VALUE = "1" VALUE = "2" VALUE = "3" VALUE = "4" VALUE = "5" VALUE = "6" VALUE = "7" VALUE = "8" VALUE = "9"

This attribute	Is used to describe these characteristics	And may contain these values
PATTERN	One of 64 valid patterns from the fill pattern palette in FileMaker Pro—for background fills in layout parts, fields, text, and field labels, and for borders of fields, field labels, and text. See "Finding values for patterns and colors" on page 5-9.	PATTERN = "1" PATTERN = "47" PATTERN = "64" PATTERN = "NONE" PATTERN = "SOLID" PATTERN = "LTGRAY" PATTERN = "GRAY" PATTERN = "DKGRAY"
SIDES	One to four sides on a field's border. To describe all four sides, you can combine all four values.	VALUE="TOP" VALUE="BOTTOM" VALUE="LEFT" VALUE="RIGHT" Or any combination, such as: VALUE= "TOP BOTTOM LEFT RIGHT" VALUE= "LEFT TOP"
SIZE (for the FONT element)	The point size for a font. Any valid point size can be specified. If a font size is unavailable on the computer or for a particular font, FileMaker Pro will substitute the closest size.	SIZE = "36" SIZE = "12" SIZE = "9"

This attribute	Is used to describe these characteristics	And may contain these values
SIZE (for the PEN element)	<p>Thickness in pixels for the outline of text blocks, field labels, and field borders.</p> <p>The value for NONE is “0” and the value for HAIRLINE is “-1.”</p> <p>When applied to field borders, this pen size also applies to the line width of an EFFECT attribute (such as DROPSHADOW) and must have a value greater than zero.</p>	<p>SIZE = “0”</p> <p>SIZE = “-1”</p> <p>SIZE = “1” through</p> <p>SIZE = “8”</p> <p>SIZE = “12”</p>
STYLE	<p>Character styles for text in fields, text blocks, and field labels. More than one style can be specified, separated by commas or spaces.</p> <p>No error checking is done for contradicting styles, such as UPPERCASE and LOWERCASE.</p> <p>The PLAIN style value overrides all other style values.</p> <p>STRIKEOUT and STRIKETHRU values are the same.</p>	<p>STYLE = “PLAIN”</p> <p>STYLE = “BOLD”</p> <p>STYLE = “ITALIC”</p> <p>STYLE = “STRIKEOUT”</p> <p>STYLE = “STRIKETHRU”</p> <p>STYLE = “SMALLCAPS”</p> <p>STYLE = “UNDERLINE”</p> <p>STYLE = “WORDUNDERLINE”</p> <p>STYLE = “DBLUNDERLINE”</p> <p>STYLE = “UPPERCASE”</p> <p>STYLE = “LOWERCASE”</p> <p>STYLE = “TITLECASE”</p> <p>STYLE = “SUPERSCRIP”</p> <p>STYLE = “SUBSCRIPT”</p> <p>STYLE = “CONDENSE”</p> <p>STYLE = “EXTEND”</p> <p>STYLE = “ITALIC, BOLD, SMALLCAPS”</p>

Specifying default values for themes

FileMaker Pro uses default values to replace attributes that are invalid or missing. For each theme listed in a theme file, you can specify whether the default values are determined by the current layout settings (which change when a user changes them) or by standard layout values (the same values that FileMaker Pro uses when creating a file for the first time).

```
<THEMEDEFAULT VALUE="CURRENT"/>
```

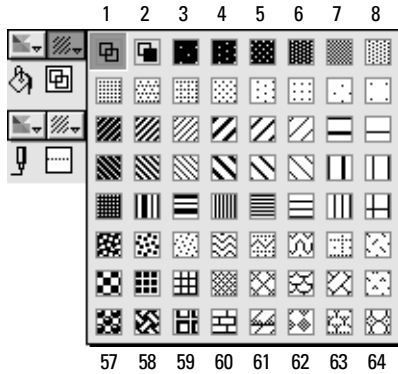
```
<THEMEDEFAULT VALUE="STANDARD"/>
```

If you don’t specify a value for the THEMEDEFAULT element in the theme, FileMaker Pro will use standard layout values by default.

Finding values for patterns and colors

The values for the patterns in the FileMaker Pro pattern palette are numbered consecutively—starting with the top row and counting from left to right, where the value for the top left pattern in the palette is 1, the value for the next pattern to the right is 2, and so on. Five patterns in the first row can also be defined with words: NONE (= 1), SOLID (= 2), DKGRAY (= 6), GRAY (= 7), and LTGRAY (= 8).

Note The first pattern (value = 1) is transparent and the second pattern (value = 2) is solid. For objects with a color fill, be sure to use the solid pattern.



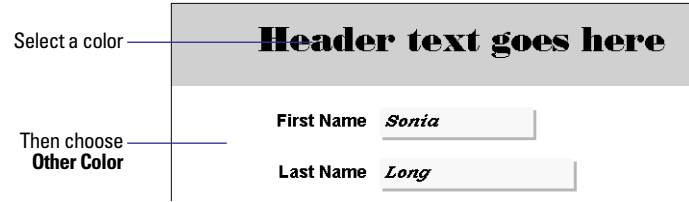
Pattern attribute values begin at the top left corner of the fill pattern palette with number 1 and end at the bottom right corner with number 64

Themes use 6-digit color hexadecimal (hex) values to describe colors (such as #CC9966), which can be found in most graphics programs that use a color palette.

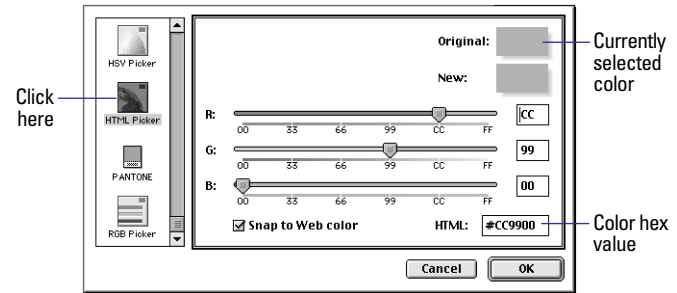
Themes should use web-safe colors for databases that will be accessed by multiple platforms, displayed on monitors with varying resolutions, or displayed on a network. However, when your databases will be displayed on a single platform or at a high resolution, the full RGB color spectrum gives you a much larger and richer color set.

To find the color hex value for a color in the FileMaker Pro color palette (a web-safe color palette):

1. In Layout mode, select a color and then choose **Other Color** from the bottom of the color palette.



2. Select the HTML Picker icon on the left side of the dialog box and read the hex value for the selected color in the HTML box on the right side.



The hex value for the selected RGB color is #CC9900

For more information about using patterns and colors in a layout, see “Customizing layouts,” in the *FileMaker Pro User's Guide* or see FileMaker Pro Help.

Adding comments to your document

You can add additional information to your XML theme documents by enclosing the information in comment tags:

```
<!-- my comment here -->
```

FileMaker Pro will ignore any unknown (but syntactically correct) XML elements you may choose to include. This allows your theme files to be backward and forward compatible with other versions of FileMaker Pro.

Checking your theme document for errors

FileMaker Pro cannot parse an XML theme document that is not well-formed, and it does not validate the XML in your documents. If one required item is missing or wrong, FileMaker Pro will ignore the entire document.

Here is a list of things to check for if your new layout themes don't appear in the New Layout/Report assistant as expected:

- The theme filename has the .fth extension.
- The theme file is in text format.
- The theme file is located in the Themes folder inside the FileMaker Developer application folder.
- All required elements are there, including their start and end tags:


```
<?xml version="1.0" standalone="yes" ?>
<FMTHEMES>
<FMTHEME>
  <THEMENAME VALUE="Purple and White Screen" />
</FMTHEME>
</FMTHEMES>
```
- All elements are complete—there are no missing attributes, values, quotation marks, start tags, or end tags.
- All values are enclosed in quotation marks (“value”)—there are no missing opening or closing quotation marks, and there are no missing values (no blank quotation marks “”).
- All elements and attributes are spelled correctly.
- All attribute values are spelled correctly and are valid.
- Every single-line element ends with />.
- All multi-line elements are spelled correctly and their start and end tags match, for example, <BODYPART> and </BODYPART>.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 6

Using the FileMaker Developer Tool

FileMaker Developer provides a utility application called the *FileMaker Developer Tool* that lets you:

- bind your database files into a stand-alone runtime database solution that does not require FileMaker Pro in order to be used on a computer
- rename all of your database files and automatically update the internal links to related files and scripts
- display your database files in Kiosk mode
- add a script to the Help menu that displays a custom Help layout or file from any layout in the database solution
- add a custom script to an About menu command so you can display a special layout screen about your solution
- change the name of the Scripts menu
- permanently prevent users from modifying the design or structure of your databases by removing access to certain menu commands
- add or remove the FileMaker Pro filename extension to your files

For information about what you need to do to prepare your database files before using the Developer Tool, see chapter 4, “Creating a database solution.”

About the solution examples

FileMaker Developer includes a stand-alone runtime solution example and kiosk solution example, created with the Developer Tool.

The runtime database solution is an example of a poster catalog, with images, prices, catalog numbers and so on. The kiosk solution is an example of a fast food ordering interface for the World Class Burger Company.

These example solutions are on the *FileMaker Developer* CD in the \Developer Extras\FileMaker, Inc\Examples folder. To see the runtime solution, on the CD double-click ..\Runtime Solution\Runtime Solution.exe (Windows), or ..\Runtime Solution\Sample Solution (Mac). To see the kiosk example, open the ..\Kiosk Solution\Menu.fp5 file in FileMaker Pro.



Primary file of the Runtime Solution Example

Using the FileMaker Developer Tool


Use the Developer Tool to create a new set of files, customize them, and enclose them inside a new solution folder.

Note You must open and save database files from versions of FileMaker Pro earlier than 5.0 before you can use them with the Developer Tool.

To use the FileMaker Developer Tool:

1. Close all of your database files that you are going to customize.
2. If necessary, install the FileMaker Developer Tool.

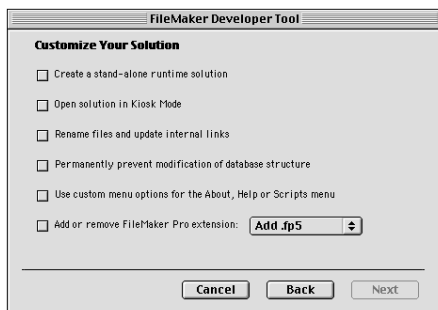
(See chapter 2, “Installing FileMaker Developer in Windows,” or chapter 3, “Installing FileMaker Developer in the Mac OS.”)

3. Double-click the  file icon to start the FileMaker Developer Tool application.



4. Click Next.

5. On the Customize Your Solution screen, select one or more options in the list.



What you select determines which screen the Developer Tool displays next.

Select this option	To create a copy of your database files and do this
Create a stand-alone runtime solution	Create a runtime database solution that includes a FileMaker Pro runtime application and the bound database files. The following menu commands become unavailable in the runtime application: Define Fields, Define Value Lists, Define Relationships, Access Privileges, Layout Mode, and ScriptMaker.
Open solution in Kiosk Mode	However, these commands can still be accessed by opening the runtime file in FileMaker Pro unless you select Permanently prevent modification of database structure below (with the exception of Define Value Lists, which is always available).
Rename files and update internal links	Create a solution that displays your databases on a full screen without the menu bar or window controls.
Permanently prevent modification of database structure	Create a database solution with new filenames. When you change the filenames, all of the internal reference links (filespecs) to related files and scripts are automatically updated.
Use custom menu options for the About, Help, or Scripts menu	Create a database solution with the following menu commands permanently unavailable in FileMaker Pro: Define Fields, Define Relationships, Access Privileges, Layout Mode, and ScriptMaker.
	Important Once removed, these menu commands cannot be made available again for the custom solution.
	Create a solution that includes a custom About or Help menu command or a different name for the Scripts menu. When you select this option, the Developer Tool presents a screen on which you can specify script names for the new menu commands that will display your custom About and Help layouts, and a new name for the Scripts menu.
	Note This option is not available if you select Open solution in Kiosk mode.

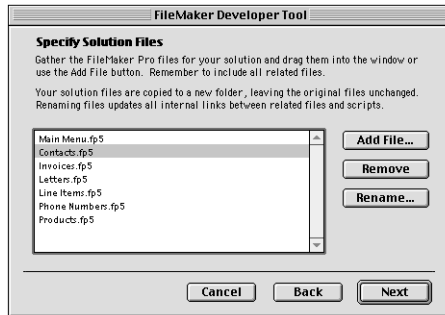
Select this option

Add or remove FileMaker Pro extension

To create a copy of your database files and do this

Add or remove the .fp5 filename extension to associate files with the FileMaker Pro application. The .fp5 extension is useful for database solutions that will be used on Windows machines, and is not necessary for solutions that will only be used on Mac OS machines.

Note This option is not available if you select Create a stand-alone runtime solution.

6. Click Next.

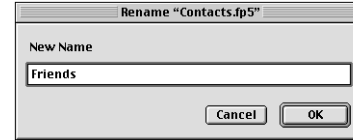
7. Drag the database files that you want to customize into the box on the Specify Solution Files screen or click **Add File** to locate and add each file individually.

The Developer Tool will create a copy of every file that is listed in this box.

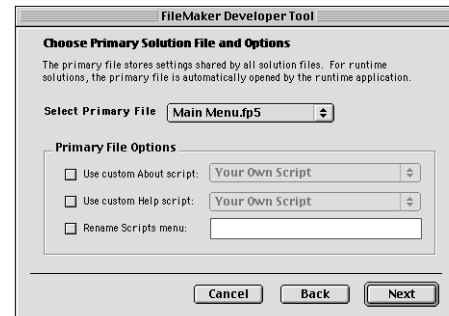
Note Make sure to specify every file that's related to the solution, so that all internal links between files can be updated. You can specify as many files as you like, but FileMaker Pro (and FileMaker Pro runtime applications) can only open up to 50 files at once.

8. If you want to use different names for the final solution files, select each file in the list and click **Rename**. In the Rename dialog box, type a name for the new file and click **OK**.

Note You do not have to type in a filename extension. However, if you want the .fp5 extension, you must type it here or select the **Add or remove FileMaker Pro extension option** (in step 5) and choose **Add .fp5** from the pop-up menu.

**9. Click Next.**

The next screen that appears depends on which options you selected on the **Customize Your Solution** screen (step 5). If you selected **Create a stand-alone runtime solution**, **Open solution in Kiosk mode**, or **Use custom menu options for the About, Help, or Scripts menu**, the **Choose Primary Solution File and Options** screen appears. If you did not select any of these options, then skip to step 21.

**10. Choose the primary file from the pop-up menu.**

The primary file stores the custom settings and is the database that users should open first. If you're binding your files into a runtime database solution, the runtime application will open the primary file first. Use this file for navigation buttons or scripts to other auxiliary files, an About layout, a custom Help layout or file, and to quit the application.

11. If desired, select **Use custom About script**, and choose the script name from the pop-up menu.

The script name will appear as an **About** menu command in the **Help** menu (Windows) or the Apple (🍏) menu (Mac OS). In Mac OS X, the script name appears in the **Application** menu.

See "Adding a custom script to an About menu command" on page 6-11.

12. If desired, select **Use custom Help script**, and choose the script name from the pop-up menu.

The script name will appear as a menu command in the **Help** menu. See "Adding a custom Help script command to the Help menu" on page 6-13.

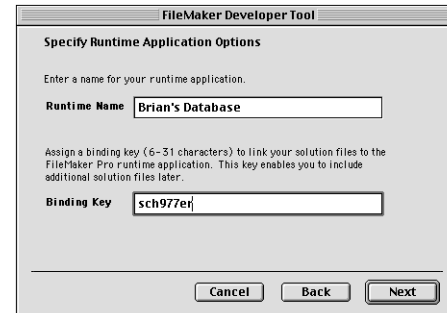
13. If you want the **Scripts** menu to be called something different, select **Rename Scripts menu** and type the new name in the box.

See "Renaming the Scripts menu" on page 6-14.

14. Click **Next**.

The next screen that appears depends on which options you selected on the **Customize Your Solution** screen (step 5). If you did not select **Create a stand-alone runtime solution**, then skip to step 21.

15. Type a name for your runtime application in the **Runtime Name** box.



The runtime name is used for the runtime application filename and can also be used for the folder name that contains the runtime database solution files.

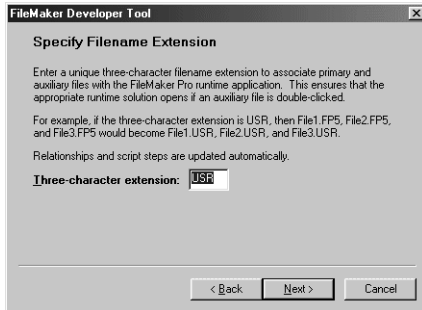
See "Specifying a runtime solution name and binding key" on page 6-7.

16. Type a key between 6 and 31 characters long in the **Binding Key** box.

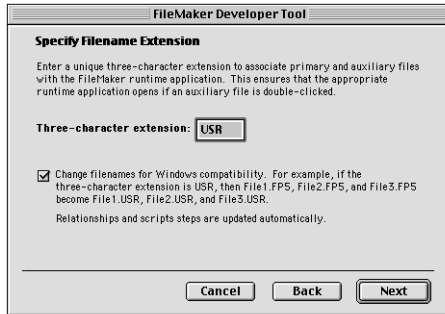
The binding key links the runtime application to the database files and ensures that the bound files will only open in the appropriate runtime application. You can use this binding key later to add updated files to the runtime database solution.

See "Specifying a runtime solution name and binding key" on page 6-7.

17. Click **Next**.



Select Filename Extension screen on Windows machines



Select Filename Extension screen on Mac OS machines

18. Type a unique set of three characters in the Three-Character Extension box that are used to associate the filenames with the runtime application.

See “Assigning a three-character filename extension” on page 6-8.

19. **Mac OS** If you plan on using the database files in the runtime solution on Windows machines, select **Change filenames for Windows compatibility**.

The Developer Tool will append the three-character extension onto each filename, making the files cross-platform compatible.

20. Click Next.



21. On the Specify Location for Solution Folder screen, click **Browse** (Windows) or **Select** (Mac OS) to specify the location for the runtime database solution.

The name of the new solution folder is determined by either the name of the primary file or the name of the runtime solution.

22. Select the location and click **OK** (Windows) or click **Choose** (Mac OS).

23. Click **Finish**.

The FileMaker Developer Tool creates a copy of all the database files, customizes them according to the options you selected, and places all the files in a new solution folder.

If you selected the option to create a runtime database solution, the Developer Tool binds all of the database files to a new runtime application file and stores the settings within the primary database file.

In addition, the Developer Tool creates a text file containing all the options you selected for your solution, which you can use with the Developer Tool again. See “Saving your settings in the Developer Tool” on page 6-14.

If an error occurs during the binding, a message displays and the error is logged in the Error Log.txt file:

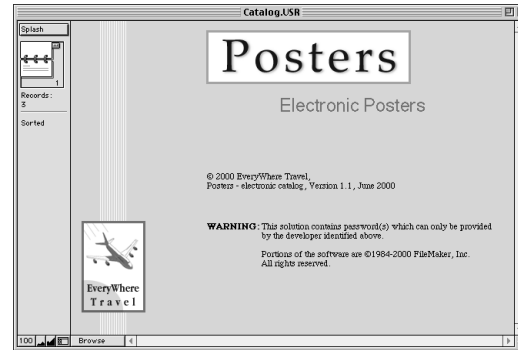
Folder	Platform
FileMaker Developer 6\Developer Tool	Windows
FileMaker Developer 6\FileMaker Developer Tool	Mac OS
FileMaker Developer 6\FileMaker Developer Tool	Mac OS X

Note Windows 2000 and Windows NT users with limited access privileges may not be able to write to the error log file in the Developer Tool folder. Log in with administrator privileges or install the FileMaker Developer Tool to a location outside of the Program Files folder.

Binding your databases into a runtime database solution

Use the FileMaker Developer Tool to bind your database files into a stand-alone runtime database solution that users can access without running FileMaker Pro on their machines. The Developer Tool creates a copy of your files, and binds the primary database file and all auxiliary files to a runtime application with a name that you specify. Users double-click the runtime application to open the bound primary file.

Note Runtime database solutions cannot be published over a network, the Internet, or an intranet unless you use FileMaker Pro instead of the runtime application. For a complete list of the differences between the runtime application and FileMaker Pro, see appendix A, “Feature comparison of the runtime application and FileMaker Pro.”



A startup script in the primary file displays this splash screen layout

Your database files should be completely developed and thoroughly tested before you bind them into a runtime database solution. See chapter 4, “Creating a database solution.”

For information about what users need to use your runtime database solution, see chapter 7, “Distributing FileMaker Pro runtime database solutions.”

Procedure for binding files

To bind your database files into a FileMaker Pro runtime database solution, do the following:

1. Close all of your database files that you are going to bind and make a backup copy.

Note Although the Developer Tool processes a duplicate copy of a file instead of the original, it's always a good idea to make a backup copy of your original files before beginning.

2. Start the FileMaker Developer Tool application and click Next.

3. Select Create a stand-alone runtime solution, select other options as desired, and click Next.

4. Specify the database files that you want to bind, rename them if desired, and click **Next** to go to the next screen in the runtime binding process.

5. Select a primary file.

All of the settings for binding are stored in the primary file. The primary file opens automatically when a user double-clicks the runtime application.

6. Select **Use custom About script** and choose the script's name from the pop-up menu.

FileMaker Pro runtime database solutions are required to have an About layout that gives information about your company and where users can go for technical support. See “Customizing the About, Help, and Scripts menus” on page 6-11 for information.

7. Select other options as desired, and click **Next**.

8. Specify a unique name and binding key for the solution and click **Next**.

Note The binding key must be between 6 and 31 characters, and is case-sensitive.

For more information, see “Specifying a runtime solution name and binding key” next.

9. Specify a unique three-character filename extension to associate all of the database files with the runtime application and click **Next**.

You must specify an extension. For files that will be used on Mac OS machines, the Developer Tool inserts the letter “F” after the first character of the extension to indicate a creator code. You should register the extension as a creator type with Apple Computer. See “Assigning a three-character filename extension” on page 6-8.

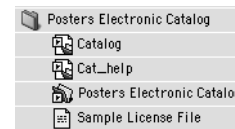
10. Specify the location for the runtime database solution files, rename the folder if desired, and click **Finish** to bind the files.

To quickly repeat the binding process, see “Saving your settings in the Developer Tool” on page 6-14.

For information on bundling the necessary files and delivering your new runtime database solution to your users, see chapter 7, “Distributing FileMaker Pro runtime database solutions.”

Specifying a runtime solution name and binding key

The runtime name that you specify in the Developer Tool is used for the name of the runtime application and can also be used for the name of the new solution folder that contains the bound runtime database solution files.



The runtime name is used for the folder and application names in this solution

The binding key is a code that the FileMaker Developer Tool uses during the binding process to internally link the files with the runtime application. If you need to add auxiliary files later to the existing runtime database solution, rebind the files using the same key.

When developing a cross-platform solution, use the same key when you bind the solution in Windows and in the Mac OS. Also, keep in mind that the binding key is case-sensitive on both Windows and Mac OS machines.

Note Use a binding key you'll remember and record it in a safe place. (You can do this by saving the Developer Tool Options file—see “Saving your settings in the Developer Tool” on page 6-14 for information.) Otherwise, if you forget your binding key and want to change a runtime database solution, you'll need to rebind all of the database files using a new binding key and then redistribute the entire solution, including a new runtime application.

Assigning a three-character filename extension

The three-character filename extension associates all of the solution files with the FileMaker Pro runtime application. If a user has more than one runtime database solution on a machine, the filename extension together with the binding key will ensure that the correct runtime application is started when a solution file is double-clicked.



The .pec filename extension associates the files with the runtime application

Use an extension that is unique to your users' computer systems.

Assigning the extension for Windows solutions

The three-character extension registers your runtime application with the Windows operating system. The extension is used by Windows to determine which application starts when you double-click a solution file. The Developer Tool appends the extension to all database filenames in the runtime database solution during the binding process.

Assigning the extension for Mac OS solutions

In the Mac OS, the three-character extension becomes the creator code for the runtime application. The creator code must be unique to ensure that the Mac OS Finder can determine which application created each document. The creator code is stored in the solution files and in the runtime application.

Because creator codes are four characters, the Developer Tool inserts an uppercase "F" after the first character. For example, the default three-character extension "USR" becomes the "UFSR" creator code. Creator codes are case-sensitive.

Note Creator codes should be registered with Apple Computer to verify that the creator code you choose is unique. If the creator code is not unique, solution files might not open with the appropriate runtime application. You may use the USR three-character extension because FileMaker, Inc. has registered the UFSR creator code with Apple Computer, Inc. Contact Apple Developer Support or visit their web site at www.apple.com to register any other creator codes.

Conflicts with non-unique filename extensions

If the three-character extension is not unique, it might cause registry (Windows) or desktop (Mac OS) conflicts. For example, if you use the .fp5 extension for your runtime database solution and your users have FileMaker Pro installed on their hard disks, all of their FileMaker Pro document icons (and the application icon in the Mac OS) will change to the runtime icons. Additionally, FileMaker Pro documents will no longer automatically open the FileMaker Pro application.

Windows To restore the document icons to the original FileMaker Pro document icon:

1. Discard the runtime application.
2. Open a document in the FileMaker Pro application, then close it and exit the application.
3. Restart your computer.

Mac OS To restore the document and application icons to the original FileMaker Pro icon:

1. Discard the runtime application.
2. Rebuild the desktop file: Hold down Option + \mathcal{R} and choose Special > Restart. Continue holding down the keys while the computer is restarting. Then, when it displays an alert dialog box, click OK to rebuild the desktop.

Note No matter what the filename extensions are, runtime database files can still be opened in the FileMaker Pro application. To prevent users from modifying your runtime database solutions, create passwords for specific access privileges or select the **Permanently prevent modification of database structure** option in the Developer Tool before you bind the files into a runtime database solution. (See “Removing design access to your databases” on page 6-11 and “Protecting your database solution files” on page 4-7 for information.)

Binding files for cross-platform solutions

If your solution will be used in Windows, bind it using the Developer Tool for Windows. If your solution will be used on the Mac OS, bind it using the Developer Tool for Mac OS. If you’re creating a cross-platform solution to be used on both Windows and the Mac OS, bind the solution files twice: first using FileMaker Developer Tool for Windows, and then using FileMaker Developer Tool for Mac OS. Use the same binding key on both platforms. Also, remember that binding keys are case-sensitive.

When you’re binding database files on a Mac OS machine for a cross-platform runtime database solution, select the **Change filenames for Windows compatibility** option. (See step 18 in “Using the FileMaker Developer Tool” on page 6-5.)

The Developer Tool automatically updates all files to use the three-character extension that you specify on this screen and appends the extension to the filenames. Internal file references used in relationships, scripts, and external value lists are updated to interact with the new filenames.

See “Design tips for cross-platform solutions” on page 4-11 for additional cross-platform information.

Modifying bound runtime files

You can open a bound runtime file in FileMaker Pro to make modifications to it, for example, to access the **Define Value Lists** menu command. However, if you selected the **Permanently prevent modification of database structure** option when you bound the files, then you can’t regain access to these menu commands: **Define Fields**, **Define Relationships**, **Access Privileges**, **Layout Mode**, and **ScriptMaker**. In this case, you’ll have to open the original database files in order to make design or structure changes in FileMaker Pro and then rebind them using the binding key that you assigned to that runtime database solution.

See “Distributing updates to your runtime database solution” on page 7-6 for more information.

Creating Kiosk-mode solutions

To display your database files in Kiosk mode, you must either bind your solution database to a stand-alone runtime application, or assign a limited access password to your solution database, and choose the Kiosk option.

To display your solution in Kiosk mode:

1. Close all of your database files that you are going to include in your Kiosk solution.
2. Start the FileMaker Developer Tool application and click **Next**.
3. Select **Open file in Kiosk mode**, select other options as desired, and click **Next**.
4. If you have already assigned a limited access password to the database solution, you can choose any other options. If you have not assigned a limited access password, you must choose **Create a stand-alone runtime solution**, in addition to any other options you want.

See “Protecting your database solution files” on page 4-7 for information about assigning passwords.

5. Specify the database files that you want to customize, rename them if desired, and click **Next**.
6. Select a primary file, select other options as desired, and click **Next** to go through the rest of the screens depending on what other options you selected.
7. Specify the location for the solution files, rename the folder if desired, and click **Finish**.

The Developer Tool creates a duplicate copy of the database files with a new Kiosk-mode interface and places them in a new folder.

Renaming your databases

Use the FileMaker Developer Tool to rename all the files in your database solution and automatically update all internal reference links (filespecs) to related files and scripts. The Developer Tool creates a duplicate copy of the database files before it renames them and updates the links.

Procedure for renaming files

To rename your database files and update reference links, do the following:

1. Close all of your database files that you are going to customize.
2. Start the FileMaker Developer Tool application and click **Next**.
3. Select **Rename files and update internal links**, and click **Next**.
4. Specify the database files that you want to customize, then select each file in the list and click **Rename**.
5. In the Rename dialog box, type a name for the new file, and click **OK**.

Note If you want the .fp5 extension to be added automatically to the new filenames, select the **Add or remove FileMaker Pro extension** option in step 3 and choose **Add .fp5** from the pop-up menu.

6. When you've finished renaming all of the specified files, click **Next**.

7. Specify the location for the files, rename the new solution folder if desired, and click **Finish**.

The Developer Tool creates a duplicate copy of the database files with the new filenames and updated internal links, and places them in a new folder.

Choosing filenames for your runtime database solution

When choosing filenames for your runtime database solutions, consider the platforms on which your runtime solution will be used so your scripts and lookups will work properly.

Platform	Filename support
Mac OS/Mac OS X	31 characters
Windows 98	130 characters, including path
Windows NT	130 characters, including path

In Windows, the FileMaker Developer Tool will not process files with pathnames longer than 130 characters. If needed, move files closer to the root level of your hard drive before opening them with the Developer Tool.

Windows filenames must not start with a space. For cross-platform compatibility, do not use the following characters in filenames: period (.), quotation mark ("), slash (/), backslash (\), brackets([]), colon (:), semicolon (;), vertical bar (|), equal sign (=), or comma (,).

Removing design access to your databases

Use the FileMaker Developer Tool to prevent users from altering the design and structure of your database files and from changing any passwords or groups that you've set up. By specifying the **Permanently prevent modification of database structure** option for your custom solution, users can't access the following menu commands in FileMaker Pro:

- File menu > Define Fields
- File menu > Define Relationships
- View menu > Layout Mode
- Scripts menu > ScriptMaker
- File menu > Access Privileges

Important Selecting this option *permanently* removes access to these menu commands for all database files in the solution, whether they're opened in a runtime application or in FileMaker Pro. This will help protect your files from password hackers. Structural elements of the files cannot be modified by anyone, including FileMaker, Inc. employees. The only way to modify field definitions, relationships, scripts, or access privileges is by returning to the original file before it was customized by the Developer Tool.

Consider the long-term needs of your users when defining access privileges. Communicate their access privileges to them clearly in the About layout and follow the rules specified by FileMaker, Inc. See “Adding a custom script to an About menu command” on page 6-11 and “Your responsibilities as a developer” on page 4-15 for information.

Steps for preventing database modification

To create a database solution that cannot be redesigned or modified, do the following:

1. Close all of your database files that you are going to customize.

2. Start the FileMaker Developer Tool application and click Next.
3. Select **Permanently prevent modification of database structure**, select other options as desired, and click **Next**.
4. Specify the database files that you want to customize, rename them if desired, and click **Next** to go through the rest of the screens depending on what other options you selected.
5. Specify the location for the solution files, rename the folder if desired, and click **Finish**.

The Developer Tool creates a duplicate copy of the database files with design access removed and places them in a new solution folder.

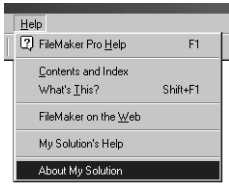
Customizing the About, Help, and Scripts menus

You can customize the menu bar in your database solution to display a custom script as a menu command in the Help menu or Apple (Ⓜ) menu (Mac OS), or to display a custom name for the Scripts menu. The custom settings are stored in the primary file of your solution.

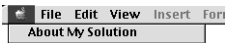
For information about designing the primary file for your solution, see chapter 4, “Creating a database solution.”

Adding a custom script to an About menu command

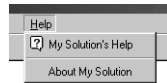
Use the FileMaker Developer Tool to add a custom script to the About menu command in FileMaker Pro or your runtime application that goes to an About layout for your solution. When the Developer Tool processes your database files, it creates a menu command named “About <your solution>” and places it in the Help menu (Windows) or Apple (Ⓜ) menu (Mac OS).



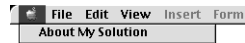
Custom About command in FileMaker Pro (Windows)



Custom About command in FileMaker Pro (Mac OS)



Custom About command in a runtime application (Windows)

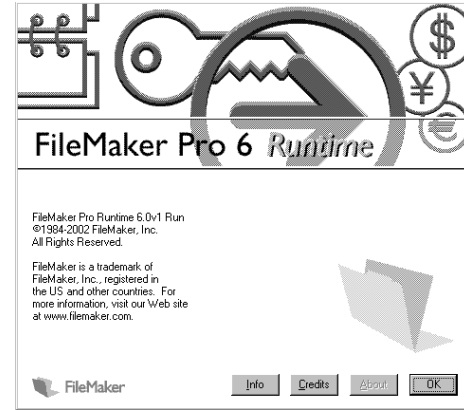


Custom About command in a runtime application (Mac OS)

In runtime applications, the custom About menu command replaces the About FileMaker Pro menu command. If you don't specify an About script when you bind your files into a runtime database solution, an About FileMaker Pro Runtime menu command is added to the Help menu (Windows) or Apple (🍏) menu (Mac OS) of the runtime application.



When users choose this menu command, the FileMaker Pro Runtime dialog box appears.



This dialog box appears by default if you don't specify a custom script for the About menu command in the runtime application

Note You are required to include an About layout in runtime database solutions that provides information about your company and where users can go for technical support. By identifying your runtime database solution with an About layout in the format required by FileMaker, Inc., FileMaker, Inc. employees are alerted not to provide technical support or issue passwords to unauthorized users attempting to open the solution. See “Abiding by the license agreement” on page 1-6 and “Your responsibilities as a developer” on page 4-15 for information.



The About layout in the Runtime Solution Example includes a button to return to the main screen

To add a custom script to an About menu command in your solution:

1. Close all of your database files that you are going to change.
2. Start the FileMaker Developer Tool application and click Next.
3. Select Use custom menu options for the About, Help, or Scripts menu, and click Next.
4. Specify the database files, and click Next.
5. Specify the primary file, select Use custom About script, and choose the script's name from the pop-up menu. Then click Next.
6. Specify the location for the new solution folder.
7. Click Finish.

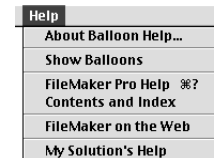
See “Creating an About layout” on page 4-8 for more information.

Adding a custom Help script command to the Help menu

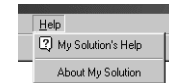
Create your own Help layout or file and a script to open it, then use the FileMaker Developer Tool to add the script name to the Help menu. In runtime applications, the custom Help script command replaces the FileMaker Pro Help command.



Custom Help command in FileMaker Pro (Windows)



Custom Help command in FileMaker Pro (Mac OS)



Custom Help command in a runtime application (Windows)



Custom Help command in a runtime application (Mac OS)

To add a custom Help script command to the Help menu in your solution:

1. Close all of your database files that you are going to change.
2. Start the FileMaker Developer Tool application and click Next.
3. Select Use custom menu options for the About, Help, or Scripts menu, and click Next.
- Note** This option is not available if you selected the Open solution in Kiosk mode option.
4. Specify the database files, and click Next.
5. Specify the primary file, select Use custom Help script, and choose the script's name from the pop-up menu. Then click Next.
6. Specify the location for the new solution folder.
7. Click Finish.

For more information, see “Creating a custom Help layout or file” on page 4-9.

Renaming the Scripts menu

Use the Developer Tool to rename the Scripts menu for your database solution. The setting is stored in the primary file of your solution and the new menu name appears in the menu bar in FileMaker Pro and in a runtime application.



Database solution with Scripts menu



Database solution with Company Reports menu

The menu name must not exceed 31 characters.

To rename the Scripts menu for your solution:

1. Close all of your database files that you are going to change.
 2. Start the FileMaker Developer Tool application and click **Next**.
 3. Select **Use custom menu options for the About, Help, or Scripts menu**, and click **Next**.
- Note** This option is not available if you selected the **Open solution in Kiosk mode** option.
4. Specify the database files, and click **Next**.
 5. Specify the primary file, select **Rename Scripts menu**, and type the new name for the menu in the box (1 to 31 characters long). Then click **Next**.

Windows To specify a keyboard accelerator, type an ampersand (&) before the character you want to use as the accelerator key. For example, type **Company Re&ports** to display the **Company Reports** menu with the letter “p” as the accelerator key.

6. Specify the location for the new solution folder.
7. Click **Finish**.

Adding the FileMaker Pro extension to database filenames

Use the FileMaker Developer Tool to quickly add the FileMaker Pro filename extension to your database files. For example, if you created your database files on a Mac OS machine but you now want the database solution to be used on Windows machines, you can add the .fp5 filename extension to your database filenames for Windows compatibility.

To add the FileMaker Pro extension to your database filenames:

1. Close all of your database files that you are going to change.
2. Start the FileMaker Developer Tool application and click **Next**.
3. Select **Add or remove FileMaker Pro extension**, choose **Add .fp5** from the pop-up menu, and click **Next**.
4. Specify the database files, and click **Next**.
5. Specify the location for the new solution folder.
6. Click **Finish**.

The Developer Tool creates a duplicate copy of the database files with the new extended filenames, updates all internal reference links, and places them in a new solution folder.

Saving your settings in the Developer Tool

You can save the settings you specify in the FileMaker Developer Tool setup screens and reuse them again. When the Developer Tool processes your specifications, it automatically creates a text file named “<Solution Name> Solution Options” and places it in the same folder that contains the Developer Tool.

Note Windows 2000 and Windows NT users with limited access privileges may not be able to write to the Solution Options file in the Developer Tool folder. Log in with administrator privileges or install the FileMaker Developer Tool to a location outside of the Program Files folder.

Using the Solution Options text file

Use the Developer Tool Solution Options file to repeat the same process on your database files with the Developer Tool.

To use the Solution Options file:

1. Drag the Solution Options text file onto the Developer Tool application file.

The Developer Tool will open to the first screen. All of the following screens contain your saved settings from before, including the names and paths of the database files.

2. Click **Next** to go through the screens and review your options.
3. Click **Finish** to process the files.

Saving a set of options

A Solution Options text file is written each time you use the Developer Tool for a database solution, and overwrites the old file of the same name. If you want, you can rename the options file to preserve a set of options — the Developer Tool will accept any filename.

Modifying the Solution Options text file

You can modify the Developer Tool Solution Options file to automatically customize the database files without displaying any setup screens. When you drag the modified text file onto the Developer Tool application file, the files are immediately processed (copied, renamed, bound, etc.) and placed in a new solution folder.

To modify the text file for auto-customizing:

1. Open the Solution Options file in a text editor.

```
Version=1
Primary File=Alice HD:Desktop Folder:DevTool Example:Relational Example:Main Menu.fp5
Secondary File=Alice HD:Desktop Folder:DevTool Example:Relational Example:Contacts.fp5
Secondary File=Alice HD:Desktop Folder:DevTool Example:Relational Example:Invoices.fp5
Secondary File=Alice HD:Desktop Folder:DevTool Example:Relational Example:Letters.fp5
Secondary File=Alice HD:Desktop Folder:DevTool Example:Relational Example:Line Items.fp5
Secondary File=Alice HD:Desktop Folder:DevTool Example:Relational Example:Phone
Numbers.fp5
Secondary File=Alice HD:Desktop Folder:DevTool Example:Relational Example:Products.fp5
Solution Name=Main Menu Solution
Binding Key=123456
Kiosk Mode=0
Remove Access=0
AddFMExtension=
Custom About=
Custom Help=
Rename Script Menu=
Extension=USR
Change Extension=1
Solution Location=Alice HD:Desktop Folder:DevTool Example:
Auto=0
```

The FileMaker Developer Tool Solutions Options file keeps a record of the solution name and the binding key that you used for that solution

2. Scroll down to the **Auto** option at the bottom of the list of options.
3. Change the value from **0** to **1**.

Now, when you drag the text file onto the Developer Tool, the customizing process is performed immediately based on the database files and other settings specified in the text file.

Tip You can use the modified Solution Options text file in your ActiveX or AppleScript scripts to help automate the binding process. For example, write a script to:

1. Clean up the database files (such as deleting example records).
2. Use the Solution Options file to bind the database files into a runtime solution.
3. Make an installer.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 7

Distributing FileMaker Pro runtime database solutions

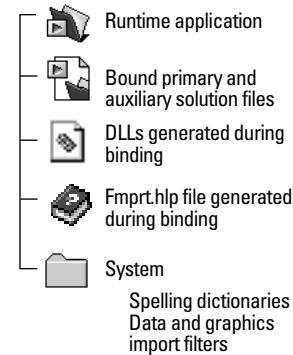
The final steps in developing your FileMaker Pro runtime database solution are to bundle all of the necessary files together, choose how you will distribute your solution—for example, on a CD-ROM disc, over a network, or on floppy disks—and provide your users with documentation for installing your solution. In addition, your documentation should include instructions for starting the runtime application and what to do if a file is damaged.

Organizing your runtime database solution components

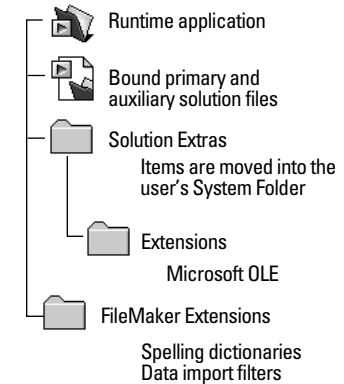
When you bind your database files into a runtime database solution, the Developer Tool creates a new solution folder and places the runtime application file and the bound primary and auxiliary database files inside it. For Windows runtime database solutions there are also several required Dynamic Link Library (DLL) files and a What’s This? Help file.

To supplement your runtime database solution, FileMaker Developer provides additional folders of files that you can include with your runtime database solutions, which are located in the FileMaker Developer 6\FileMaker Developer Tool\Runtime Files folder.

Windows solution folder



Mac OS solution folder



Example of Windows and Mac OS solution folders for distribution

Copy these files and their folders into your solution folder. When your users first start the runtime application, the files will either work immediately, or the runtime application will automatically copy them into the appropriate folders on the users’ hard disks.

Important These files and folders must not be renamed or moved from the solution folder. Your users must restart their computers after copying to the solution folder.

For information about the FMPRT.hlp file that’s generated for runtime databases on Windows systems, see “Providing What’s This? Help (Windows)” on page 4-10.

In addition to these runtime files, you will need to provide installation instructions for your users. See “Documenting the installation procedures” on page 7-5.

Generated DLL files for Windows runtime database solutions

The following table describes the Dynamic Link Libraries (DLLs) that are provided for your runtime database solutions running on Windows machines. These files are generated by the Developer Tool during the binding process and must reside in the root level of the solution folder along with the runtime application and bound database files.

Required DLL files	Description
Fmeng10.dll	Database engine
Fmcor10.dll Fmfc10.dll Fmgfx10.dll Fmint10.dll• Fmnsv14.dll Fmole10.dll • Fmqte10.dll Fmwfc10.dll• Fmml10.dll• Fmcon10.dll•	XPAL files •
Fm_usr.dll -	Resource file for the filename extension you specified (.USR by default)
ClIngenu.dll Clproof.dll	Spelling checker files•
Mfc42.dll	MFC (Microsoft Foundation Class) file•
Msvcr7.dll	MSVC runtime file•
Ctl3d32.dll	Required by MFC •
XalanDOM.dll XalanExtensions.dll• XalanSourceTree.dll• XalanTransformer.dll• xerces.dll• XercesParserLiaison.dll• XMLSupport.dll• XPath.dll• XSLT.dll•	XML/XSL files•

Important In addition to the generated DLL files, there are two required DLL files, Shfolder.dll and Comctl32.dll, that need to be installed separately on your users' hard disks. These DLLs are automatically installed by Windows NT 4.0 with Service Pack 3 and by Internet Explorer 4.0. If your users cannot find either of these DLLs on their hard disks, you'll need to instruct them to get Internet Explorer from the Microsoft web site at www.microsoft.com.

System files for Windows solutions

The following table describes the data and graphic import files and spelling dictionaries that you can include with your runtime database solution for Windows.

Copy the FileMaker Developer 6\Developer Tool\Runtime Files\System folder and its contents into the root level of your runtime database solution folder. The files should immediately work with the runtime application on the user's hard disk.

System folder and files	Description
System folder-	Copy folder and its contents into the root level of the runtime database solution folder—Contains files for graphics and data import, and four spelling dictionaries.
Fmbmp32flt	BMP graphic import filter
Fmcm32flt	CGM graphic import filter
Fmdrw32flt	DRW graphic import filter
Fmmac32flt	MacPaint graphic import filter
Fmpcx32flt	PCX graphic import filter
Fmpic32flt	PIC graphic import filter
Fmsld32flt	Lotus SLD graphic import filter
Fmtiff32flt	TIFF graphic import filter
Fmwmf32flt	Metafile graphic import filter
Dbf.imp	Used to import dBASE files

System folder and files	Description
Excel.imp	Used to import Excel files
XML.imp	Used to import XML files
Claddon.clr-	Spelling dictionary that contains computer terminology
Ukenglish.mpr-	Primary British English dictionary of over 100,000 words that the runtime application uses to check the spelling
Usenglish.mpr-	Primary US English dictionary of over 100,000 words that the runtime application uses to check the spelling
User.upr-	Spelling dictionary that stores your customized list of spelling exceptions and additions to the User dictionary

Solution Extras files for Mac OS solutions

The following table describes additional graphic import filters and extensions that you should include with your runtime database solution for Mac OS users.

Note These extension files are for Mac “Classic” OS only. They do not apply to Mac OS X.

Copy the FileMaker Developer 6\FileMaker Developer Tool\Runtime Files\Solution Extras\Extensions folder and its contents into the root level of your runtime database solution folder. When the runtime application is started, the files inside the Solution Extras\Extensions folder are automatically copied into the appropriate folders in the user’s System Folder.

Tip You can add other folders and files to the Solution Extras\Extensions folder and they will be copied to the appropriate location in the System Folder on the user’s hard disk. For example, include a Fonts folder of special fonts for your solution in the Solution Extras folder, and when the runtime application is started all fonts are copied into the Fonts folder on the user’s hard disk.

Solution Extras\Extensions files	Description
Microsoft OLE Automation	Files for Excel data importing
Microsoft OLE Extension	
Microsoft OLE Library	

FileMaker Extensions files for Mac OS solutions

The following table describes the FileMaker Extensions files you can distribute with your runtime database solution for Mac OS. With these files, users will be able to import data from certain versions of dBASE and Excel databases into your runtime database solution. They will also have access to four spelling dictionaries.

Copy the FileMaker Extensions folder and its contents from the FileMaker Developer Tool\Runtime Files folder into the root level of your runtime database solution folder. When the runtime application is started, the files should work immediately.

FileMaker Extensions folder and files	Description
FileMaker Extensions folder	Place in same folder as runtime application to enable data importing and spelling dictionaries
DBF	Used to import dBASE files
Excel	Used to import Excel files
XML	Used to import XML files
Claddon.clr	Spelling dictionary that contains computer terminology

FileMaker Extensions folder and files	Description
Ukenglish.mpr-	Primary British English dictionary of over 100,000 words that the runtime application uses to check the spelling
User.upr-	Spelling dictionary that stores your customized list of spelling exceptions and additions to the User dictionary
Usenglish.mpr-	Primary US English dictionary of over 100,000 words that the runtime application uses to check the spelling

Choosing the distribution method

After you have organized the files that comprise your solution, you need to decide how your users will install them. You can distribute compressed files on multiple floppy disks, or distribute your bundled solution on a CD, over a network, or using other media, such as LS-120 or Zip disks.

Requirements for distributing on floppy disks

The FileMaker Pro runtime application is too big to fit on a single floppy disk. If you plan to distribute your runtime database solution on floppy disks, you'll need to use a compression utility or custom installation program, which compresses and stores the runtime application and bound solution files on several disks. Your users will need the instructions and software to decompress the runtime database solution on their hard disk.

Using a custom installation program

If your runtime database solution is large, or if you plan to distribute the files on floppy disks, consider using a custom installation program. Configuring a custom installation application to automatically install runtime database solution files may require more engineering than using a compression utility, but usually offers more professional results.

Here are some custom installation applications that you might want to use:

- MindVision Installer VISE (Windows and Mac OS) by MindVision, Inc. (www.mindvision.com)
- InstallShield (Windows) or InstallShield Java Edition by InstallShield Software Corporation (www.installshield.com)
- StuffIt InstallerMaker (Mac OS) by Aladdin Systems (www.aladdinsys.com)

Using a compression utility program

If your runtime database solution is small, you might consider a compression utility application rather than a custom installation application. To compress files, use a utility such as:

- WinZip for Windows (Windows 98, Windows NT, Windows 2000, Windows ME) by WinZip Computing, Inc. (www.winzip.com)
- StuffIt Deluxe for Mac OS (the StuffIt Expander decompression utility is available for Windows and Mac OS) by Aladdin Systems (www.aladdinsys.com)

What your users need

In order to run your FileMaker Pro runtime database solution, your users will need the same minimum equipment and software required by the FileMaker Pro application (see “Hardware and software requirements” on page 1-2). In addition, your users will need instructions for installing and starting your solution, and information about how to recover damaged files.

Sharing your solution over a network

Because the runtime application does not support FileMaker Pro file sharing (peer-to-peer or client/server networking), users cannot share your runtime database solution over a network unless they access the files using the FileMaker Pro application on their machines.

For optimal performance, they can host the solution files using FileMaker Server.

For information about the FileMaker Server and FileMaker Pro products, and information about volume license sales, see the FileMaker, Inc. web site at www.filemaker.com.

Documenting the installation procedures

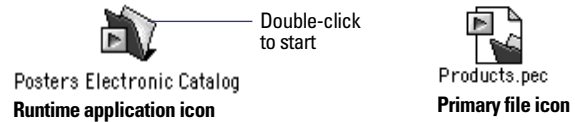
You'll need to provide instructions to your users for how to install your runtime database solution. Here's a list of things you should document:

- Provide written instructions for copying or installing your solution to your user's hard disk.
- Include software and instructions specifying how your users can decompress your solution files.
- Include information about the minimum equipment and software requirements.

For suggestions on other information to include with your runtime database solution, see "Including printed documentation" on page 4-10.

Starting your runtime database solution

The first time users double-click the runtime application icon, the runtime application will auto-register and move appropriate items into the System folder on their machines.



In Windows, the three-character filename extension associated with the solution will not be registered by the operating system until the runtime application has been started. If a primary or auxiliary solution file is double-clicked before the runtime application has registered the extension, the runtime application won't be found.

Important Your users should start your solution by double-clicking the runtime application icon, not the primary file icon. Double-clicking the icons for the primary or auxiliary files might result in errors, depending on whether there are other copies of the runtime application on the hard disk. If your users have more than one solution on their computers with the same three-character extension and they double-click the icon for the primary file, the most recently installed runtime application is opened, which may not be the correct application for your solution's primary file.

Each time the runtime application is opened, it looks for the primary file that has been bound to it. If the primary file can't be found, the user is asked to locate the primary file.

Caution your users that they should not rename the primary or auxiliary solution files. If they do, relationships and external scripts may not work properly. They can rename the runtime application after it has been installed.

Note When you make a change to your solution, you should make sure that your users can import their data into your updated solution. Include a script attached to a button to make it easy for your users to import their data into the new solution files. For more information, see "Converting and upgrading solution files" on page 4-16.

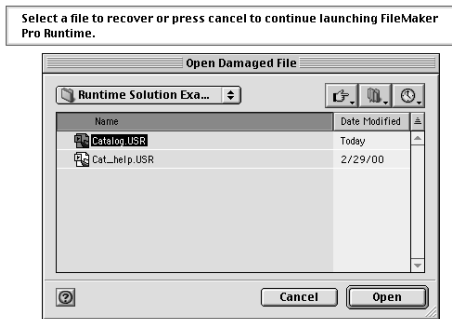
Recovering damaged files

Power failures, hardware problems, or other factors can damage a FileMaker Pro file. If your database solution becomes damaged, your users need to recover the damaged file. When the runtime application discovers a damaged file, a dialog box appears, telling the user to contact the developer.

Once you know which file is damaged, you can recover it—however, the Recover command does not appear in the File menu of the runtime application.

To recover a damaged file:

- On Windows machines, press Ctrl+Shift while double-clicking the runtime application icon. Hold the keys down until you see the Recover File dialog box.
- On Mac OS machines, press Option+⌘ while double-clicking the runtime application icon. Hold the keys down until you see the Recover File dialog box.



Recovering a runtime database solution file

During the recovery process, the FileMaker Pro runtime application:

- creates a new file
- renames any damaged file by adding Old to the end of the filenames (for example, Contact Manager is renamed to Contact Manager Old).
- gives the repaired file the original name

If users experience unusual behavior in the recovered files, they should revert to a backup copy that was made before the file became corrupt, or contact you for technical assistance.

In your printed documentation, you should tell your users what to do once a file has been recovered. Tell your users to:

1. Recover the damaged solution file.
2. Open the recovered solution file in the runtime application.
3. Save a compressed copy — choose File menu > Save a Copy As. In the dialog box, choose Compressed Copy (Smaller) from the Save A (Windows) or Type (Mac OS) drop-down list, name the file, and click Save.

Give the compressed file the same name as the original filename.

Distributing updates to your runtime database solution

If you make feature enhancements or modifications to the primary file of your runtime database solution, you can distribute the updated file to your users without needing to rebind it. If you change the name of the primary file, however, you'll need to rebind the file and distribute a new version of the runtime application along with the updated file. To distribute new or updated auxiliary files for your runtime database solution, you need to bind them first using the original binding key.

If you forget the original binding key for your runtime database solution and want to update or add a file, you'll need to rebind all of the database files with a new binding key and redistribute the entire solution.

To distribute an updated primary file:

- 1.** Open the original primary file (before it was bound) in FileMaker Pro.
- 2.** Make the changes to the primary file. If necessary, create an Import script so users can import their existing data into the new primary file. (See "Importing data into an upgraded runtime database solution" on page 4-17 for information.)

As long as the binding key and the filename have not changed, you don't need to rebind the primary file.

- 3.** Send your users a copy of the new primary file with instructions to replace the old primary file in the runtime database solution folder.

To distribute a new or updated auxiliary file:

- 1.** In FileMaker Pro, create the new auxiliary file or open the original auxiliary file (before it was bound) and make changes as desired.
- 2.** If necessary, create an Import script so users can import their existing data into the new file. (See "Importing data into an upgraded runtime database solution" on page 4-17 for information.)
- 3.** Use the Developer Tool to rebind all of the files in the runtime database solution and include the new or updated auxiliary file. (See "Procedure for binding files" on page 6-6 for information.)

Use the same binding key that you used for the primary file. Remember that the binding key is case-sensitive.

- 4.** Send your users a copy of the new or updated auxiliary file along with instructions to place it in the runtime database solution folder, replacing the old file if appropriate.

As long as the binding key has not changed, you don't need to redistribute the runtime application or other solution files.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 8

Publishing your database on the Web

The FileMaker Pro Web Companion plug-in makes it possible for you to publish your database on the Internet or an intranet in several different ways, giving you more choices and control over the design and functionality of your web pages.

You can publish your database with:

- custom web publishing using XML
- FileMaker Pro database-aware Java applets using the FileMaker JDBC Driver
- custom web publishing using CDML
- FileMaker Pro Instant Web Publishing
- a custom home page using Instant Web Publishing
- custom web publishing using a database layout
- static web publishing (exporting data into an HTML table)

When you serve your databases and your instant or custom web pages via the Web Companion, users can access your databases from their web browsers using a simple URL.

This chapter gives general information on setting up and using the Web Companion for creating custom web pages.

Types of web publishing

Custom web publishing with XML

You can use the Web Companion to generate data from your FileMaker Pro databases into Extensible Markup Language (XML) documents. With these XML documents, you can (for example) use JavaScript and the W3C Document Object Model to dynamically manipulate data after it has been downloaded from your database. Many of the actions (such as searching) can be performed without the need to reconnect to the database, making the web user's interaction with the database happen much faster.

For information on custom publishing your database with XML, see chapter 10, "Using FileMaker Pro XML to deliver your data on the Web."

For a list of XML resources and examples, and an overview of what XML is and how it can be used with FileMaker Pro, visit "FileMaker XML Central," on the FileMaker support pages at www.filemaker.com.

Custom web publishing with JDBC

The *FileMaker Developer* CD provides a JDBC (Java database connectivity) API-compatible driver that allows you to create FileMaker Pro database-aware Java applets for your web site using any Rapid Application Development (RAD) tool.

With a FileMaker Pro database-aware Java applet, you can make your databases function more like they are being used in FileMaker Pro rather than in a web browser.

For information, see chapter 11, "Using JDBC to deliver your data."

For a list of JDBC resources, see the product support pages on the FileMaker, Inc. web site at www.filemaker.com. As a shortcut, in FileMaker Pro, choose Help menu > FileMaker on the Web.

Custom web publishing with CDML

The FileMaker Pro Web Companion lets you publish your database with custom web pages using a proprietary markup language called CDML. Included with FileMaker Pro are all the tools, templates, and examples you need to create your own custom web pages using CDML. For information, see chapter 9, “Custom web publishing using CDML.”

Instant Web Publishing

When you use FileMaker Pro Instant Web Publishing, the Web Companion generates predesigned HTML forms with HREF links to your database. For general information on web publishing and using the Instant Web Publishing feature, see chapter 14, “Publishing databases on the Web,” in the *FileMaker Pro User's Guide* or see FileMaker Pro Help.

Other ways to create custom web sites for your data

If you want, you can create a custom home page to go with your instant web pages instead of using the built-in FileMaker Pro Instant Web Publishing home page. See “Creating a custom home page” on page 8-5, “Creating a custom home page for Instant Web Publishing” on page 8-6, and “Creating a custom web site using a database layout” on page 8-7.

Static web publishing with HTML

You can also publish your data on static web pages if you don't need dynamic web access to your database. See “Exporting data to a static HTML page” on page 8-16.

Using the FileMaker Pro Web Companion

The Web Companion is a plug-in that acts as a Common Gateway Interface (CGI) application for handling interactions between FileMaker Pro and your web browser. The Web Companion also functions as a web server by providing static files (such as HTML pages and images) to the web browser.

Web users access your database either by accessing the IP address of the computer running FileMaker Pro with their browser (which takes them to the home page) or by clicking an HREF link that contains a specific CGI request for FileMaker Pro. The Web Companion then sends via HTTP (Hypertext Transfer Protocol) either the default home page or the web page specified in the FileMaker CGI request.

The Web Companion in the Developer edition of FileMaker Pro can only serve to a maximum of ten IP addresses in a 12-hour period, as indicated by the IP Guest Limit of 10 in the Web Companion Configuration dialog box.

If desired, you can set up your computer for testing without a constant connection to the Internet or an intranet. For information, see “Testing your site without a network connection” on page 8-17.

For general information on the Web Companion and about connecting to the Internet or an intranet, see chapter 14, “Publishing databases on the Web” in the *FileMaker Pro User's Guide*.

Requirements for web access

The host computer must have a copy of FileMaker Pro serving the databases on the Web (preferably with a full-time, constant connection to the Internet or your intranet). The Web Companion must be enabled in FileMaker Pro.

There are two folders that can be used to store files to be served to the Web; the Web folder, and the cdml_format_files folder, both located at the root level of the FileMaker Pro folder. Your custom web pages, CDML format files, and XSLT style sheets can be stored in the Web folder in order for the Web Companion to serve them on the Web. Though the Web folder allows web users to view the source code of files placed within it, the cdml_format_files folder prevents web users from directly viewing the source code of any files placed within it. This allows the user to see the HTML generated after processing the CDML format file, while preventing them from seeing the CDML tags in the unprocessed source file. See the Web Security.pdf for more information on using the cdml_format_files folder (FileMaker Developer 6\Web Security\Web Security.pdf).

Your databases should not be inside either of these folders. They only need to be open in FileMaker Pro and shared via the Web Companion.

Note You can keep your site folders, web pages, and databases in a different folder anywhere on your hard drive. To do this, replace the Web folder inside the FileMaker Developer folder with a shortcut/alias named “Web”.

For information and tips on providing security for your databases on the Web, see the WebSecurity.pdf document in the FileMaker Developer Web Security folder.

Enabling the Web Companion

You only need to enable the FileMaker Pro Web Companion plug-in once. FileMaker Pro will attempt to connect to a network in order to enable the Web Companion — if you do not have a network connection but want to enable the Web Companion anyway, see “Testing your site without a network connection” on page 8-17.

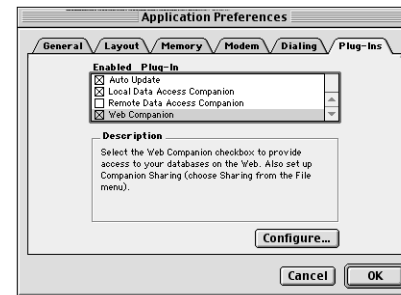
To enable the Web Companion in Windows or Mac OS:

1. In FileMaker Pro, choose Edit menu > Preferences > Application.

Mac OS X: choose FileMaker Developer application menu > Preferences > Application.

2. In the Application Preferences dialog box, click the Plug-Ins tab.

3. Select the Web Companion checkbox to enable the Web Companion plug-in.



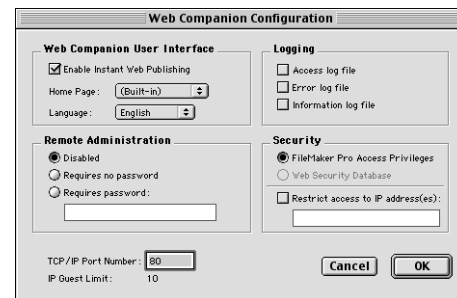
Enable the Web Companion in the Application Preferences dialog box

4. Select Web Companion and click Configure to set configuration options, or click OK.

Setting Web Companion configuration options

After you’ve enabled the Web Companion, follow these steps to select various configuration options:

1. On the Plug-Ins tab in the Application Preferences dialog box, select Web Companion and click Configure.



Example of the Web Companion configured for Instant Web Publishing

2. In the Web Companion Configuration dialog box, choose an HTML file from the Home Page list so the Web Companion will automatically display it when web users enter the IP address.

The [Built-in] option displays the “FileMaker Pro Instant Web Publishing” home page by default. All other HTML files that are located in the root level of the Web folder appear in this list.

See “Creating a custom home page” on page 8-5 for more information.

3. For custom web publishing, deselect Enable Instant Web Publishing.

4. If desired, choose a language from the Language pop-up menu for using localized texts in Instant Web Publishing navigation.

This choice will not affect the language of your data.

Note The Language setting can also be used with the [FMP-CurrentAction], [FMP-FindOpItem], or [FMP-SortOrderItem] CDML replacement tags in your custom web pages. See “Using an encoding parameter with a CDML replacement tag” on page 9-12 for information.

5. If you want, select one or more Logging options: Access Log File, Error Log File, and Information Log File.

For information, see “Web Companion support for Internet media types” on page 8-13.

6. Select a Remote Administration option.

If you want to remotely access the Web folder from a different computer, for example, to upload or download files using HTTP Put and Get commands or to change settings in the Web Security Database, select Requires Password and enter a password in the box. (If it doesn't matter who has access to the Web folder and everything inside it, select Requires no password.)

For more information, see “Opening password-protected databases remotely” on page 8-18.

7. Select a Security option.

FileMaker Pro Access Privileges is selected by default. For general information about setting access privileges in FileMaker Pro, see chapter 9, “Protecting databases with passwords and groups,” in the *FileMaker Pro User's Guide*.

In FileMaker Pro, you can secure data on a record-by-record basis using access privileges. For more information, see the Web Security.pdf file in the FileMaker Developer 6 Web Security folder or see FileMaker Pro Help.

The Web Security Database option is not available when the Web Security.fp5 database is not open. For information on the Web Security database, see the Web Security.pdf file.

8. If desired, select Restrict access to IP address(es) and type the IP addresses of the computers that are allowed to access the Web folder, the web pages served by the Web Companion, and your databases.

You can enter multiple IP addresses separated by commas and use an asterisk as a wildcard for all addresses beginning with the specified numbers. For example, 1.2.3.4,5.6.7.8,3.5.* indicates two IP addresses and all addresses that begin with “3.5.”

A computer's IP address is determined by the network administrator (for an intranet) or an Internet service provider (ISP) account.

9. Specify a TCP/IP port number.

By default, web browsers use the TCP/IP port number 80 to communicate with the web server. If that port is in use, you can use any number between 1024 and 65535 or the port number 591, which is registered by FileMaker, Inc. with the Internet Assigned Numbers Authority (IANA) for use with the FileMaker Pro Web Companion.

Mac OS X To use port numbers below 1024 on Mac OS X machines, you'll need your Mac OS X administrator name and password. See “Enabling the FileMaker Pro Web Companion (Mac OS X)” on page 3-5.

10. Click OK to close the Web Companion Configuration dialog box.

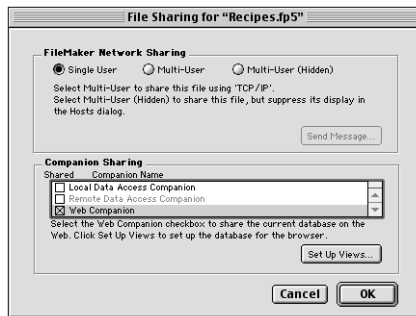
11. Click OK again to close the Application Preferences dialog box.

Sharing the database via the Web

Each database that you're publishing on the Web must be open and shared via the Web Companion.

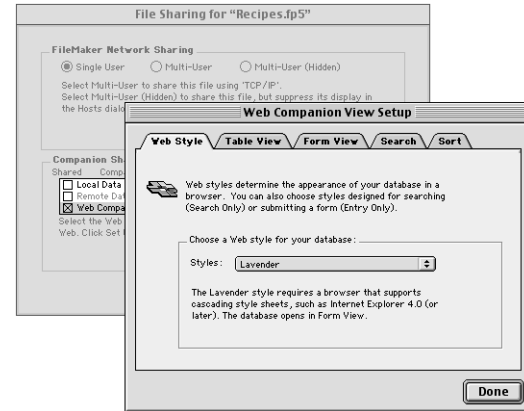
To share a database on the Web:

1. In FileMaker Pro, choose File menu > Open and open the database.
2. Choose File menu > Sharing.
3. Select the Web Companion checkbox.



If the Web Companion option is unavailable (dimmed), you need to enable the Web Companion. See “Enabling the Web Companion” on page 8-3.

4. If you're using Instant Web Publishing for this database, click Set Up Views and choose a web style and layouts for each view (instant web page).



For Instant Web Publishing, choose a web style and layouts for each view (instant web page)

5. Click Done to close the Web Companion View Setup dialog box.
6. Click OK to close the File Sharing dialog box.

Creating a custom home page

You can set the Web Companion to open a custom home page by default rather than the built-in home page (the FileMaker Pro Instant Web Publishing home page) — whether you're using Instant Web Publishing or custom web publishing with CDML or XML.

When web users enter the IP address of the host computer in their web browsers, the Web Companion will serve either the built-in home page used for Instant Web Publishing, any web page named “default.htm,” “default.html,” “index.htm,” or “index.html” that is located in the root level of the Web folder or in a site folder within it, or the custom home page you specify in the Web Companion Configuration dialog box (which must be located in the root level of the Web folder).

If you're hosting multiple sites each with its own home page, you can have each home page named "default.htm" or "index.htm" inside each site folder within the Web folder, and the Web Companion will display them when web users enter the name of the site folder after the IP address.

`http://17.17.17.17/guest_book`

If you want, create a link to the IP address of the site folder—otherwise, let users know so they can type it in their web browsers.

Note You can enter "localhost" instead of the IP address when FileMaker Pro and all the files are on your computer. See "Testing your site without a network connection" on page 8-17.

`http://localhost`

`http://localhost/guest_book`

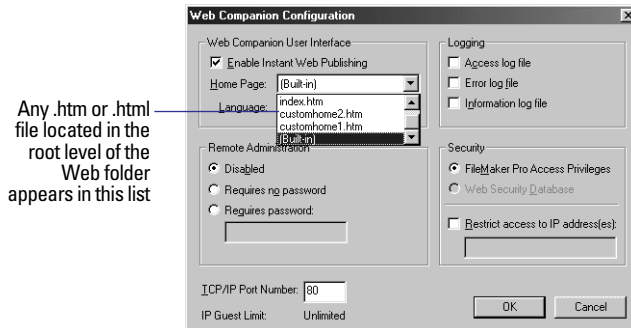
Specifying a custom home page as the default

To specify the custom home page as the default in the Web Companion Configuration dialog box:

1. Make sure the custom home page is located in the root level of the Web folder.

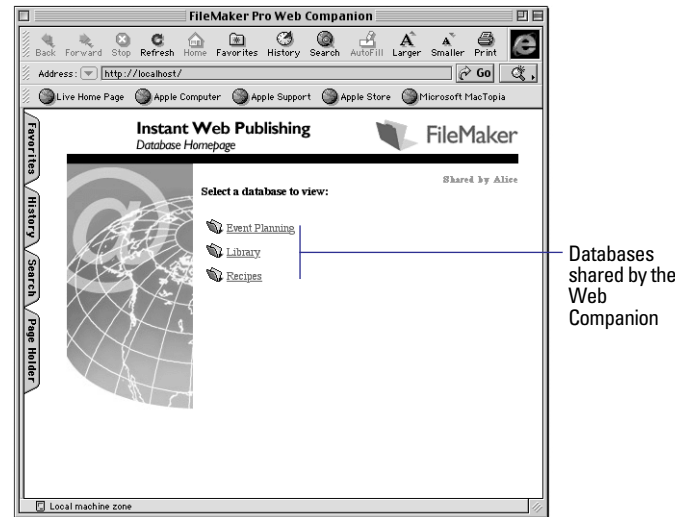
The custom home page can be named anything but must be in HTML format (with the .htm or .html filename extension).

2. In the Web Companion Configuration dialog box, choose your custom home page from the Home Page list.



Creating a custom home page for Instant Web Publishing

The simplest and quickest way to publish your FileMaker database on the Web is to let FileMaker Pro Instant Web Publishing design your web pages for you. You can create your own web page to replace the built-in FileMaker Pro Instant Web Publishing home page and still use the instant web pages generated by the Web Companion. Your new custom home page can contain anything else you want to include for your web site, such as web graphics, movies, and Macromedia Flash animations.



Example of the built-in Instant Web Publishing home page

The built-in home page for Instant Web Publishing displays a list of all the open FileMaker Pro databases that are shared via the Web Companion. Each database in the list is linked to a FileMaker CGI request for displaying the database records in different views (instant web pages), such as Form View.

About the FileMaker WebPortal object

With the Web Companion in FileMaker Pro, you can access the elements of the Instant Web Publishing home page (such as database names or the URL to access a Form View of a database) as separate JavaScript objects and extract data from them to build your own custom home page.

To access elements of the Instant Web Publishing home page, you need the following FileMaker CGI request:

```
<SCRIPT LANGUAGE="JavaScript" SRC="FMPro?-webportal">
</SCRIPT>
```

Note If you are using a Netscape web browser, you must specify JavaScript 1.4.

This HTML statement places the JavaScript object called WebPortal inside a window object in your web page. The window.webPortal object contains the following subobjects:

```
webPortal.databases = Array of <databaseObject>
webPortal.userName = Name of current FileMaker Pro user
```

Each <databaseObject> in the array contains the following:

databaseName	The string displayed for the link in the Instant Web Publishing home page
defaultURL	The default URL for opening the database in the browser based on settings made in the Web Companion View Setup dialog box or the Document Preferences dialog box
formViewURL	The URL for opening the database in the Form View instant web page
tableViewURL	The URL for opening the database in the Table View instant web page
searchViewURL	The URL for opening the database in the Search View instant web page
newViewURL	The URL for opening the database in the New Record View instant web page

Note A URL for the default Form View for a database can be used with FileMaker WebPortal objects.

A Custom Web Portal example with commented code is included in the Custom Workgroup Portal folder on the *FileMaker Developer CD*.

Overview of setting up a custom home page for Instant Web Publishing

To create a custom home page using JavaScript:

1. Create an HTML file for your web page using a text editor or HTML editing program.
2. Include a FileMaker CGI request for the FileMaker WebPortal object:


```
<SCRIPT LANGUAGE="JavaScript" SRC="FMPro?-webportal">
</SCRIPT>
```
3. As desired, use scripting to write HTML and text to the document.
4. Save the file with the .htm or .html filename extension and place it in the root level of the Web folder (inside the FileMaker Developer folder).
5. In the Web Companion Configuration dialog box, select **Enable Instant Web Publishing**, choose your custom web page from the Home Page list, and click OK.

Creating a custom web site using a database layout

Features in FileMaker Pro 6 enable you to design your own page layouts for Instant Web Publishing in Layout mode, and then display the layouts in the web browser.

First, you create a startup script to hide the Instant Web Publishing interface. Then you create buttons in the layouts to navigate the web site and perform database functions. To bypass the Instant Web Publishing home page, you use a custom home page that contains a redirect statement for opening the database layout in a particular instant web page.

Overview of using a database layout as the Instant Web Publishing home page

To create a custom home page that uses a database layout for the interface:

- 1.** In Layout mode in FileMaker Pro, design the layout for your custom home page. Create script buttons for every type of interaction you want the web user to be able to do on this page. (See “Using script buttons in Instant Web Publishing” on page 8-8.)
- 2.** Create a startup script that hides the Instant Web Publishing interface. (See “Suppressing the Instant Web Publishing interface” on page 8-11.)
- 3.** Enable the Web Companion and configure it for Instant Web Publishing.
- 4.** Set up file sharing for the database through the Web Companion (see “Sharing the database via the Web” on page 8-5), and click **Set Up Views** in the File Sharing dialog box.
- 5.** In the Web Companion View Setup dialog box, choose either Soft Gray, Lavender, Wheat, or Blue and Gold 1 for the web style. Then specify the layout you want used for each view (instant web page).
- 6.** Choose **Edit menu > Preferences > Document**.
Mac OS X: Choose **FileMaker Developer application menu > Preferences > Document**
- 7.** Set up FileMaker Pro to switch to the layout you created in step 1 and perform the startup script you created in step 2 when the database opens.

When the database is opened in the web browser, the layout that you select for the **Switch to Layout** option in the Document Preferences dialog box overrides all layouts selected in the Web Companion View Setup dialog box for file sharing.

8. Create a web page with a redirect statement that bypasses the built-in Instant Web Publishing home page and displays your custom layout in the browser. (See “Bypassing the Instant Web Publishing home page” on page 8-12.)

9. Configure the Web Companion to open the web page as the default home page. (See “Specifying a custom home page as the default” on page 8-6.)

Now when you enter your computer’s IP address or “localhost” in the web browser, the Web Companion will display the database layout for your custom home page in the browser window.

Using script buttons in Instant Web Publishing

You can provide special script buttons in your FileMaker Pro layout to work with Instant Web Publishing. When web users click on a button in the browser, the script’s URL is sent to the Web Companion as a FileMaker CGI request.

If the script is a single script step, an **onClick** JavaScript event handler is executed and a URL is generated containing the current state information in the browser and information from the script step extracted by the Web Companion.

If the script contains multiple script steps, state information from the first three supported steps is extracted to construct a JavaScript state object (`scriptState`) that encapsulates the result of the executed script. The resulting information is passed to the JavaScript runtime application (Instant Web Publishing), which interprets the state object, builds the resulting URL, and sends the CGI request to the Web Companion.

Note For information about FileMaker CGI requests made in custom web publishing, see “Generating FileMaker Pro CGI requests using CDML” on page 9-3 and “Generating FileMaker Pro CGI requests for an XML document” on page 10-8.

Requirements for Instant Web Publishing buttons

A button that you’re using in a layout for Instant Web Publishing may have a single valid script step attached to it or a script containing 1 to 3 valid script steps.

If you’re using multiple text and graphic objects for a button, the script or script step must be attached to the topmost object in the group. Create the text and graphic elements first, group them, and then attach the script to the group.

Single script steps supported for Instant Web Publishing

In FileMaker Pro, the Web Companion supports the following single script steps for buttons used in Instant Web Publishing layouts.

FileMaker script step	CGI request	Description
Open [<Document name>]	Open database in browser window	Equivalent to opening the database from the Instant Web Publishing home page. The database must be specified as a script parameter, and it must be open in FileMaker Pro.

FileMaker script step	CGI request	Description
Open URL [<url>]	Set window location to the specified URL	Use this with a text field or calculation field with a text result to construct target URLs. The URL must be complete (e.g. include http://) and can go to another web site or contain a FileMaker CGI request.
Go to Layout [<Layout Name>]	Go to specified layout	This will not affect other current parameters for location. The generated URL link is based on the default URL of the database plus any settings made for startup script and layout, and specified record in a relationship. <i>Also supported in a multi-step button.</i>
Go to Related Record [<Relationship Name>]	Go to specified record in a related database	The related database must be open and shared via the Web Companion, and the specified record must be available when the script is performed.
Go to Record/Request/Page [n]	Go to record number	The record number can be specified by a constant, by a field value, or from a JavaScript prompt. <i>Also supported in a multi-step button.</i>
Go to Record/Request/Page [First]	Go to first record	<i>Also supported in a multi-step button.</i>

FileMaker script step	CGI request	Description
Go to Record/Request/Page [Last]	Go to last record	<i>Also supported in a multi-step button.</i>
Go to Record/Request/Page [Previous]	Go to previous record	<i>Also supported in a multi-step button.</i>
Go to Record/Request/Page [Next]	Go to next record	<i>Also supported in a multi-step button.</i>
Go to Field []	Go to Edit Record view	Displays the specified field in the Edit Record page (without a blinking insertion point). <i>Also supported in a multi-step button.</i>
New Record/Request []	Go to New Record view	<i>Also supported in a multi-step button.</i>
Enter Browse Mode []	Go to Form View	Allows the web user to go from an Edit Record, Search, or New Record page to Form View without submitting a CGI request. <i>Also supported in a multi-step button.</i>
Enter Find Mode []	Go to Search page	<i>Also supported in a multi-step button.</i>
Show All Records	Find all records	<i>Also supported in a multi-step button.</i>
Perform Find	Submit -find request	This does not restore a -find request that has been saved with the script. <i>Also supported in a multi-step button.</i>

FileMaker script step	CGI request	Description
Exit Record	Submit form	This submits -edit record, -new record, and -find requests. <i>Also supported in a multi-step button.</i>
Sort []	Go to Sort page	<i>Also supported in a multi-step button.</i>
Delete Record/Request []	Delete record with confirmation alert	
View As [View As Table]	View current layout in CSS table	
View As [View As Form]	View current layout in CSS form	
View As [View As List]	View current layout in CSS table	This is the same as the View As [View As Table] script step
View As [Cycle]	Cycle between Form and Table views	
Open Help	Open HTML help	This opens the built-in help for Instant Web Publishing, called "FileMaker Pro Web Companion Help" in a new browser window.

Note When you use the Go to Related Record script step with Instant Web Publishing, the sort order of the related database is based on the Sort View settings in the Web Companion View Setup dialog box (not on the sort order specified by the relationship). The found set of records is determined by the relationship — only related records are in the current found set when the script is performed.

Support for scripts with multiple script steps

The Web Companion supports 1 to 3 script steps in a script button used for Instant Web Publishing. (Any steps after 3 valid script steps are ignored.) The script must include a change of mode, layout, or current record. The script may also include the submission of a form containing either a -find request or an edited record. The following script steps can be used in multi-step scripts:

Go to Record
Go to Layout
Go to Field
Sort
Enter Find Mode
Enter Browse Mode
New Record
Show All Records
Exit Record
Perform Find

If the first script step in the script is not one of these supported script steps, then it is handled as a single script step. If it is not supported for Instant Web Publishing, then the script is not generated for the button. If a script contains both supported and unsupported steps, then parsing of the script will cease as soon as the first unsupported step is encountered.

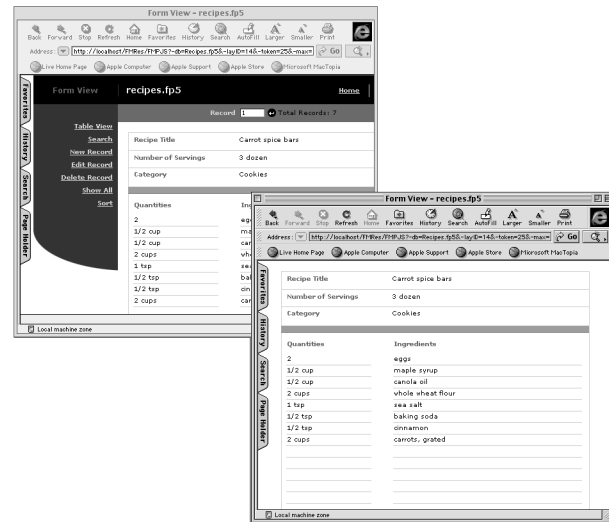
Suppressing the Instant Web Publishing interface

You can use a startup script to suppress the automatic page layouts and navigation controls of Instant Web Publishing in the browser. When web users click a link on the Instant Web Publishing home page, your database layout appears instead of the built-in layout of the instant web page. (To display a database layout instead of the Instant Web Publishing home page, see “Bypassing the Instant Web Publishing home page” on page 8-12.)

When you hide the Instant Web Publishing interface, you must specify one of the following web styles that use cascading style sheets:

- Soft Gray
- Lavender
- Wheat
- Blue and Gold 1

The other web styles don’t work with hiding the Instant Web Publishing interface. For information about web styles, see “Choosing a web style” in chapter 14 of the *FileMaker Pro User’s Guide* or see FileMaker Pro Help.



Example of a web page before and after the Instant Web Publishing interface is hidden

You only need the Toggle Status Area [Hide] script step in your startup script to hide the Instant Web Publishing interface. In addition, you can combine the Toggle Status Area [Hide] script step with one of the following script steps in the startup script:

- Enter Browse Mode: Form View
- Enter Find Mode: Search page
- New Record: New Record View
- View As [View as Table]: Table View

The Freeze Window, Set User Capture, and Refresh Window script steps can appear before the supported steps, but they will be ignored.

To hide the Instant Web Publishing interface:

1. Choose **Scripts > ScriptMaker** and type a name for the new script in the Script Name text box. Then click **Create**.
2. In the Script Definition dialog box, click **Clear All**, and select **Toggle Status Area**. Choose **Hide** from the **Specify** pop-up menu to add the parameter to the script step. Then click **OK**.
3. Click **Done** to close the Define Scripts dialog box.
4. Mac OS, Windows: Choose **Edit > Preferences > Document**.
Mac OS X: Choose **FileMaker Developer menu > Preferences > Document**.
5. In the Document Preferences dialog box, select the checkbox for **Perform script when opening the database**, and choose the script you named in step 1 from the pop-up menu.
6. Click **OK**.

For more information, see “Defining scripts” in chapter 10 and “Setting document preferences” in appendix A of the *FileMaker Pro User's Guide* or see FileMaker Pro Help.

Bypassing the Instant Web Publishing home page

You can bypass the built-in Instant Web Publishing home page so that the database layout you've created appears as the default home page in the web browser. You do this by writing a redirect statement in an HTML file that includes a FileMaker CGI request and then designating the file as the default home page in FileMaker Pro.

For the FileMaker CGI request, you'll need to know the URL of the view (instant web page) that you want the database layout to appear in. You can get this from the browser window by displaying the database in the Instant Web Publishing home page and moving the cursor over the link or clicking the link to go to the view.

To bypass the Instant Web Publishing home page:

1. Create an HTML file that contains a redirect statement to your database layout.
2. Save the HTML file with the .htm or .html extension and place it in the Web folder.
3. In the Web Companion Configuration dialog box, specify the HTML file to be the default home page. (See “Setting Web Companion configuration options” on page 8-3.)

For example, the following redirect statement contains a FileMaker CGI request for layout ID number 3 in the “MyCustomUI.fp5” database to open in the Form View instant web page (formvwcss.htm).

```
<HTML>
<BODY>
    <SCRIPT Language="JavaScript">
        window.location = "/FMRes/FMPJS?-db=MyCustomUI.fp5&
        -layID=3&-token=25&-max=1&-format=formvwcss.htm&
        -mode=browse&-findall"
    </SCRIPT>
</BODY>
```

Note In the Internet Explorer 4.5 for Mac OS browser window, you must allocate at least 6 MB of memory to the web browser in order to display the database layout.

Here's another example of a redirect statement that displays the database layout in the browser window.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/1999/REC-html1401-19991224/loose.dtd">
<HTML>
<HEAD>
<META HTTP-EQUIV="Refresh" Content="0; URL=/FMRes/FMPJS?
    -db=MyCustomUI.fp5&-layID=3&-token=25&-max=1&
    -format=formvwcss.htm&-mode=browse&-findall">
</HEAD>
<BODY>
</BODY>
</HTML>
```

Note Layout ID numbers are determined by the original creation order of all the layouts created for the database.

Format filenames for instant web pages

The following table lists the instant web pages and their format filenames as they apply to those web styles that use cascading style sheets.

Instant web page	Format filename
Form View	FormVwCSS.htm
Table View	TableVwCSS.htm
Search	SearchCSS.htm
New Record	NewCSS.htm
Edit Record	EditCSS.htm
Delete Record	DeleteCSS.htm

Instant web page	Format filename
Sort	SortCSS.htm
Error	Err.htm

Note When you suppress the Instant Web Publishing controls, your users will be completely dependent on your buttons and scripts to manage your database solutions in a browser. You can simulate a “home” link by creating a button to perform a script composed of the single step, Open URL [No dialog, “http:”]. (Note there is only one “/” after “http:”.) This statement can’t be attached directly to the button itself, but must be included in a script performed by the button.

Web Companion support for Internet media types

The Web Companion supports current MIME (Multipurpose Internet Mail Extensions) types registered for the Internet. The Web Companion can also serve other types of media files besides HTML to the web browser with the correct MIME type.

For example, you might want to serve WML (Wireless Markup Language) documents from your web site. Browsers with the appropriate plug-in would be able to display the file in the browser window.

For information about the Internet media type registry, go to <http://ftp.iana.org/in-notes/iana/assignments/media-types/>.

Monitoring your site

The FileMaker Pro Web Companion generates three types of log files that you can use for gathering information about web users who visit your site:

- access.log
- error.log
- info.log

For information on enabling log files, see “Setting Web Companion configuration options” on page 8-3.

In addition, the Web Companion provides several external functions for monitoring activity with your databases, which can be used in your calculation fields and scripts.

Using the access.log file

The access.log file keeps a record of every time someone accesses the Web Companion from a web browser and lists the hits in NCSA/CERN-compatible Common Log Format.

```

127.0.0.1 - - [13/Mar/2001:14:50:31 -0800] "GET
/FMRes/FMPJS?-db=recipes.fp5&-layid=1&-format=newcss.htm&-max=1&-skip=4
&-token.0=25&-mode=new&-lop=and&-findall HTTP/1.1" 200 3283
127.0.0.1 - - [13/Mar/2001:14:50:32 -0800] "GET
/FMRes/FMPPro?-db=recipes.fp5&-layid=1&-format=newcss.htm&-max=1&-skip=4
&-token.0=25&-mode=new&-lop=and&-findall&-iupdata HTTP/1.1" 200 24609
127.0.0.1 - - [13/Mar/2001:14:50:47 -0800] "POST
/FMRes/FMPPro?-db=recipes.fp5&-layid=1&-token.0=25&-token.1=&-skip=6&-na
x=1&-format=Formuvcss.htm&-mode=browse&-new= HTTP/1.1" 200 5536
127.0.0.1 - - [13/Mar/2001:14:50:47 -0800] "GET
/FMRes/FMPPro?-db=recipes.fp5&-layid=1&-token.0=25&-token.1=&-skip=6&-na
x=1&-format=Formuvcss.htm&-mode=browse&-new=&-find&-iupdata HTTP/1.1"
200 11464

```

When you enable the Access Log File option in the Web Companion Configuration dialog box, the Web Companion generates an access.log file and places it in the root level of the FileMaker Pro folder.

Every time a web user accesses your database, the Web Companion continuously adds entries to the access.log file.

Note Neither the entries nor the file are automatically deleted, and so the file may become very large. To save hard disk space on your host computer, consider archiving the access.log file on a regular schedule.

The Common Log Format used for the access.log file is:

remotehost rfc931 authuser [date] "request" status bytes

Where	Means this
remotehost	The remote IP address or hostname
rfc931	Required for UNIX systems
authuser	The user name authenticated by the web user
[date]	The date and time of the request
"request"	The request line exactly as it came from the client
status	The HTTP status code returned to the client (for information, see the World Wide Web Consortium's web site at www.w3c.org)
bytes	The content length of the document transferred to the client

Using the error.log file

The error.log file, stored in the root level of the folder containing the database, is generated by the Web Companion whenever any unusual errors have occurred. Common errors reported to the web user, such as “Database not open,” are not recorded in the error.log file.

[08/Jun/1999:16:16:01:53 -0800] Web Security database not open.
Security disabled.

[12/Jul/1999:06:07:02 -0800] ERROR: 6. Could not find email format file.

[23/Jul/1999:11:12:38 -0800] ERROR: 12: Badly formatted URL.

Using the info.log file

The info.log file, stored in the root level of the folder for the database, contains entries generated by the [FMP-Log] CDML replacement tag. Whenever web users access FileMaker Pro from your custom CDML web page, information you've included within a [FMP-Log] tag is recorded by the Web Companion in the info.log file.

For information about the CDML replacement tags, see chapter 9, "Custom web publishing using CDML."

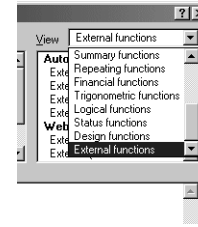
Using the Web Companion external functions

You can use the FileMaker Pro Web Companion external functions with your calculations or scripts to:

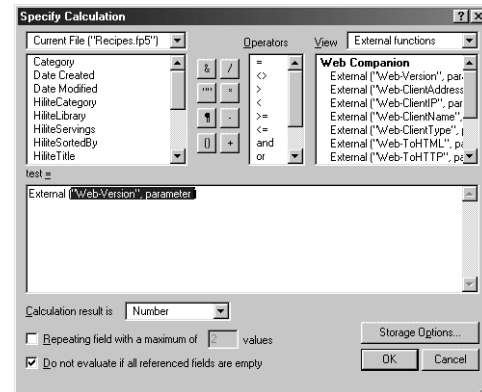
- check the version of the Web Companion
- capture information about visitors to your database
- translate information in your database to HTML or HTTP

To use a Web Companion external function in a calculation field:

1. Be sure the Web Companion is enabled. (See "Enabling the Web Companion" on page 8-3 for information.)
2. Choose File menu > Define Fields.
3. Type a name for the new calculation field in the Field Name box.
4. For Type, select Calculation, and click Create.
5. In the Specify Calculation dialog box, choose External Functions from the View pop-up menu.



6. Double-click one of the external functions in the list that begins with the function prefix "Web-" to add it to the formula box.



A formula for an external function requires the name of the external function to call and the function's parameter.

7. Replace the word "parameter" with the required parameter for the function (0, field name, or text value).

See the next table for a description of the Web Companion external functions and their parameters.

8. Continue to build the formula as desired and click OK when you're done.

9. Click Done to close the Define Fields dialog box.

External function's name and parameter	Description of external function
Web-Version, 0	Returns the version of FileMaker Pro Web Companion that loads when you open FileMaker Pro
Web-ClientAddress, 0	Returns the domain name (for example, www.filemaker.com) of a web user whose HTTP request is currently being processed by the Web Companion. Returns the web user's IP address if the domain name is not available.
Web-ClientIP, 0	Returns the IP (Internet protocol) address of the web user whose HTTP request is currently being processed by the Web Companion
Web-ClientName, 0	Returns the value that the web user types for a user name in the web browser password dialog box
Web-ClientType, 0	Returns the name and version of the web browser being used by the web user
Web-ToHTML, field name Web-ToHTML, text value	Returns the contents of the specified field or text value encoded in HTML. This is useful, for example, when you want to modify a data field with a calculation and then use it as an error page.
Web-ToHTTP, field name Web-ToHTTP, text value	Returns the contents of the specified field or text value encoded in HTTP. This is useful for fields containing data that should be handled as URLs in HREF links by the browser. For example, "Display Art" becomes "Display%20Art."

For more information, see chapter 11, "Using formulas and functions," in the *FileMaker Pro User's Guide* or see FileMaker Pro Help.

Exporting data to a static HTML page

To quickly display the existing data in your FileMaker Pro database on the Web, you can export it into an HTML table and publish your database as a static web page. To automate the exporting process on a regular basis, use a FileMaker Pro script and control the timing for when the exported table is uploaded to the web site.

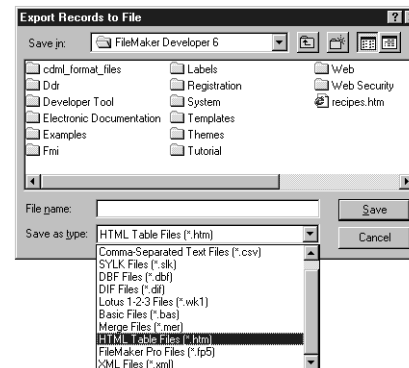
Note Images in container fields cannot be exported. To display them on your web site, place the image files in the Web folder and insert reference links to them in a text field in the database (or insert image links in the exported HTML table).

To export the data into an HTML table:

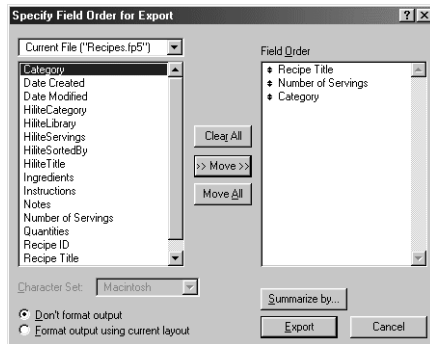
1. Open the FileMaker Pro database and create a sorted found set with the records you want to export.

If you are going to export subsummary data, include the break field in the sort order.

2. In Browse mode, choose File menu > Export Records. Type a name and select a location for the exported file. Then, choose HTML Table Files (Windows) or HTML Table (Mac OS) for the file type, and click Save.



3. In the Specify Field Order for Export dialog box, indicate how you want FileMaker Pro to export the data by selecting fields in the left box and moving them to the Field Order box on the right — in the order that you want them to appear.



4. To export a related field, choose the relationship from the pop-up menu.

The related fields for the relationship you chose are then listed in the Specify Field Order for Export dialog box.

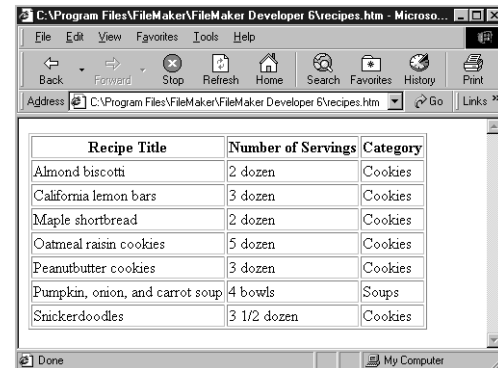
5. To export one record that subtotals multiple records, click Summarize by. Then, in the Summarize by dialog box, click to the left of the field you want to use as the break field. A check mark appears to the left of the field. (You must have set up one or more summary fields and sorted by the break field in the database.) Click OK.

In the Export Field Order for Export dialog box, FileMaker Pro creates italicized temporary export fields in the Field Order list based on the break fields you specified. For example, if you have a summary field named Count and you select the break fields Category and Unit, the export field list contains Count, *Count by Category*, and *Count by Unit*.

6. Select Don't format output if you want to ignore number, date, and time field formatting (for example, to export a number as 3.7 instead of as \$3.70).

7. Select Format output using current layout if you want to use number, date, and time field formatting as specified for the included fields on the current layout (for example, to export a number as \$3.70 instead of as 3.7). Symbols and other non-numeric values are exported as text.

8. Click Export.



FileMaker Pro saves the data in table format in an HTML file. Each field you specified becomes a column heading in the table and each record becomes a row.

Testing your site without a network connection

You can set up your computer to test the Web Companion and your web site before uploading your site files and databases to the host web server or dialing up to an Internet service provider (ISP).

To set up your computer to act as a single-machine network, enable and activate TCP/IP networking. On Windows machines, this is usually already set up for you (if you can connect to the Internet, then TCP/IP networking is active). On Mac OS machines, one way to set up TCP/IP networking is to create a new TCP/IP control panel configuration that connects via Ethernet to any IP address manually. (For information on setting up TCP/IP networking, see the documentation that came with your operating system.)

Once you've set up your computer to act as a single-machine network, you can type `http://localhost` in your web browser and the FileMaker Pro Web Companion will serve the HTML pages that are located in the Web folder as well as any open databases that are shared via the Web Companion—without connecting to the Internet or intranet.

Tip To thoroughly test your web site, click on every link that exists in your custom web pages under every possible situation, with your databases open, with (and without) any records existing in each database. Did you catch all the errors and create an error message for each of them?

Opening password-protected databases remotely

You can open and close FileMaker Pro databases from your web browser or other client application by making a `-dbopen` or `-dbclose` request to FileMaker Pro.

Note You can also open and close FileMaker Pro databases remotely by using the `DbOpen` and `DbClose` pseudo procedures with the FileMaker JDBC Driver. See “Using `DbOpen` and `DbClose` pseudo procedures” on page 11-5 for information.

The databases must be located in the Web folder and the Web Companion Configuration dialog box must have **Remote Administration** enabled. In addition, you should require a remote administration password to ensure that once databases are opened, they cannot be closed by an unauthorized user.

The Web Companion uses HTTP basic authentication to enforce web security. When a `-dbopen` request is made to FileMaker Pro, the browser or client application displays the basic user name/password dialog box where you type `admin` for the user name and the remote administration password that you specified in the Web Companion Configuration dialog box.

For databases that also have an access privileges password, you must use the `-password` parameter with the `-dbopen` request. After you enter the admin user name and remote administration password, the Web Companion checks the `-password` parameter in the request.

Tip For better security, place your databases in subfolders within the Web folder. This way, unauthorized users will not know the rest of the path even if they gain access to the Web folder.

Opening and closing databases using XML

Here is an example of making a `-dbopen` request using an XML document:

```
FMPPro?-db=secretfolder/employees.fp5&-format=-fmp_xml&-password=dbpassword&-dbopen
```

Here is an example of making a `-dbclose` request using an XML document:

```
FMPPro?-db=secretfolder/employees.fp5&-format=-fmp_xml&-dbclose
```

For more information, see “Generating FileMaker Pro CGI requests for an XML document” on page 10-8 and chapter B, “Valid names used in CGI requests for FileMaker Pro XML data.”

Opening and closing databases using CDML

To open or close databases remotely using a `-dbopen` request or `-dbclose` request (CDML variable tags), you must also specify a `-format` parameter.

If desired, use the `[FMP-CurrentError]` and `[FMP-CurrentDatabase]` tags in the format file to display the results of the request (the “current database” was successfully opened or the “current database” could not be opened because of “error #”).

Here is an example of making a `-dbopen` request:

```
FMPPro?-db=secretfolder/employees.fp5&-format=dbopen.htm&  
-password=dbpassword&-dbopen
```

Here is an example of making a `-dbclose` request:

```
FMPPro?-db=secretfolder/employees.fp5&-format=dbclose.htm&  
-dbclose
```

For more information, see “Generating FileMaker Pro CGI requests using CDML” on page 9-3.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 9

Custom web publishing using CDML

The FileMaker Pro Web Companion supports CDML, a proprietary markup language that enables your HTML pages to interact with a FileMaker Pro database.

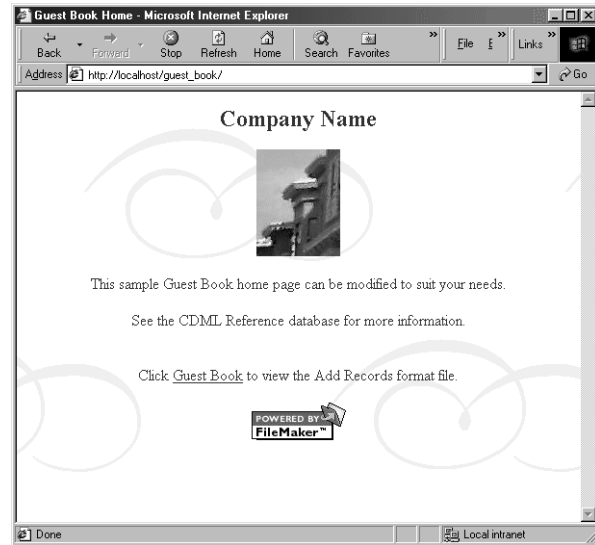
The *FileMaker Developer* CD includes:

- the CDML Tool database and template files to help you insert the CDML tags into your HTML pages (These HTML pages that contain CDML tags are referred to as *format files*.)
- the CDML Reference database, which describes each CDML tag and how it's used in a format file
- example web sites that publish databases dynamically on the Web using CDML

Tip You can also publish your FileMaker Pro databases on the Web using open standard XML or via Java applets that use the FileMaker JDBC Driver. See chapter 10, “Using FileMaker Pro XML to deliver your data on the Web” and chapter 11, “Using JDBC to deliver your data.”

About the CDML examples

FileMaker Developer provides three examples of databases published on the Web using CDML. The Guest Book example demonstrates how web users can add records to your database by “signing” in a guest book. The Employee Database example is a web site that lets users search the Employees.fp5 database and make modifications to it. The Shopping Cart example lets web users select items from a database and add them to a “shopping cart” for purchase. For information about these CDML examples, see “Planning your web site” on page 9-12.



Home page of the Guest Book example

For additional information and examples using CDML to publish FileMaker Pro databases on the Web, see the product support pages on the FileMaker, Inc. web site at www.filemaker.com. As a shortcut to the site, double-click `Go_FileMaker.html` on the *FileMaker Developer* CD.

General steps for custom web publishing using CDML

In general, to publish your FileMaker Pro databases on the Web you must have a TCP/IP connection to an intranet or Internet Service Provider, the Web Companion must be enabled, and your databases must be open and shared via the Web Companion. (For more information, see chapter 8, “Publishing your database on the Web.”)

1. Design your web site and how web users will interact with your database. See “Planning your web site” on page 9-12 for ideas.
2. Create a format file for every type of interaction web users will have with your database, for example, adding records to the database. Use the CDML Tool to add tags to your format files and use the format file templates provided by the CDML Tool as a starting point. See “Using the CDML Tool and templates” on page 9-5 for information.
3. Create a custom home page for your web site that contains a link to the first format file you want displayed and the path to the database. See “Creating a custom home page” on page 8-5 and “Generating FileMaker Pro CGI requests using CDML” on page 9-3 for information.
4. Place the format files and the custom home page in the Web folder, located inside the FileMaker Pro folder on your computer.
5. In FileMaker Pro, enable the Web Companion, open each database that you're publishing, and share it via the Web Companion.

See “Enabling the Web Companion” on page 8-3 and “Sharing the database via the Web” on page 8-5.

6. In the Web Companion Configuration dialog box, deselect **Enable Instant Web Publishing** and choose your custom home page from the **Home Page** pop-up menu if desired. (It must be located in the root level of the Web folder.)

See “Setting Web Companion configuration options” on page 8-3 for information.

7. In a web browser, enter the IP address of the computer that's hosting the database.

If you enclosed your pages within a site folder, enter the name of the folder after the IP address. Otherwise, the Web Companion will serve the home page that you specified in step 6.

Tip You can create an index page containing a link to every database site that you're publishing. Each link can contain the IP address and site folder so that users don't have to type it in their web browsers. When web users click the link, the Web Companion will serve the index.htm file located in the site folder. For example, an index page might have these three links:

```
<P><A HREF="http://17.17.17.17/guest_book">Guest Book</A></P>
<P><A HREF="http://17.17.17.17/employee_database">Employees
</A></P>
<P><A HREF="http://17.17.17.17/shopping_cart">Shopping Cart</A>
</P>
```

About CDML format files

A format file is an HTML page containing CDML tags. CDML tags are distinguished by a hyphen (-) or square brackets []. In addition, a format file may contain FileMaker Pro CGI requests in an HTML form or HREF link.

For example, every HTML form in a format file that is requesting data from the database begins with the FMPro form action and the following hidden INPUT tags.

```
<FORM ACTION="FMPro" METHOD="POST">
  < INPUT TYPE="hidden" NAME="-db" VALUE="Filename.fp5">
  < INPUT TYPE="hidden" NAME="-lay" VALUE="Layout Name">
  < INPUT TYPE="hidden" NAME="-format"
  VALUE="Filename.htm">
```

```
< INPUT TYPE="hidden" NAME="-error" VALUE="Filename.htm">
</FORM>
```

Note Format files that contain the “FMPro FMRES” form action are instant web pages generated by the Web Companion Instant Web Publishing feature. (See chapter 14, “Publishing databases on the Web,” in the *FileMaker Pro User’s Guide* for information about Instant Web Publishing or see FileMaker Pro Help.)

CDML *variable tags* are used to specify the parameters of a request:

- The names -db and -lay in this example (referred to as CDML variable tags) are used to specify the database and layout for the request.
- The -format name specifies the format file you want the Web Companion to display with the results of the database request.
- The -error name specifies the format file you want displayed in case of an error in the request. (For information on other ways to display an error page, see “Creating error messages” on page 9-11.)

For making requests to the database, the format file must contain a CDML *action tag*. For example, a Delete Record format file contains the -delete action tag in an HTML submit form button.

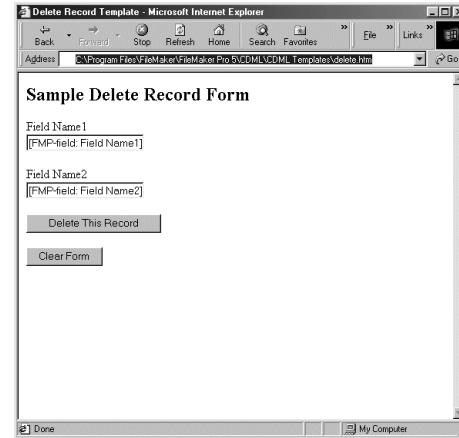
```
< INPUT TYPE="submit" NAME="-delete" VALUE="Delete this record">
```

CDML *replacement tags* act as placeholders for data. For example, if the current CDML page is based on a record with “Robert Chan” in FieldName1, the CDML tag [FMP-Field: FieldName1] is replaced with “Robert Chan”.

```
< INPUT TYPE="text" NAME="Field Name1" VALUE="[FMP-Field: Field Name1]">
```

Field Name1: Robert Chan

When a format file is displayed statically in the browser rather than as the result of a FileMaker Pro CGI request, CDML replacement tags will appear on the page.



The delete.htm template page displayed statically in the browser window

FileMaker Developer includes templates of the commonly used format files. For information, see “Using the Templates tab” on page 9-6.

Generating FileMaker Pro CGI requests using CDML

You use CDML action tags in FileMaker Pro CGI (Common Gateway Interface) commands to generate requests for data from your database.

For example, to generate a -findall request to display all employees from a database, web users might click on an HREF link containing the following FileMaker Pro CGI command:

```
FMPro?-db=Employees.fp5&-lay=FormView&-format=results.htm
&-findall
```

Or, web users might click on a submit button in an HTML form containing the FMPro form action and the following hidden INPUT elements:

```
<P><FORM ACTION="FMPro" METHOD="post">
<P><INPUT TYPE="hidden" NAME="-db" VALUE="Employees.fp5">
<P><INPUT TYPE="hidden" NAME="-lay" VALUE="FormView">
<P><INPUT TYPE="hidden" NAME="-format" VALUE="results.htm">
```

The submit button in the form contains the `-findall` request name:

```
<P><INPUT TYPE="submit" NAME="-findall" VALUE="Start Search">
```

Request names

The name of a request for CDML data is determined by the name of the CDML action tag in the request. You use CDML variable tags to specify the parameters of a request.

Use this request name (CDML action tag) To generate this request

<code>-delete</code>	Delete record
<code>-duplicate</code>	Duplicate record
<code>-edit</code>	Edit record
<code>-find</code>	Find a record
<code>-findall</code>	Find all records
<code>-findany</code>	find a random record
<code>-new</code>	New record
<code>-dbopen</code>	Open database
<code>-dbclose</code>	Close database
<code>-view</code>	Display format file

Note The `-script` variable tag (request parameter) is not designed to work with `-view` requests.

For a detailed list of the CDML action and variable tags and example syntax for using them in a FileMaker Pro CGI request, see the Tags Index in the CDML Reference database (described in “Using the Tags tab” on page 9-6).

Requests for adding records to a portal

You can use CDML to add records to a portal of related database files. When you make an `-edit` request or a `-new` request that includes data for a portal, you must specify the layout and the relationship name for the related database.

Note You can only add one record at a time to a portal, and therefore must make separate `-new` requests to add more rows to the portal.

The following is an example of a `-new` request for adding a record to a portal, where “Address:” is the name of the database relationship, and “City.0” is the related field name in the portal:

```
FMPro?-db=Employees.fp5&-lay=LayoutOne&FirstName=Sam&
LastName=Smith&Address::City.0=Seattle&-format=reply.htm&-new
```

Requests for editing multiple records in a portal

You only need to make one `-edit` request to edit multiple records in a portal. You specify each row (or record) in the portal by adding a period and a consecutive number (starting with number 1) to the end of the related field name.

The following is an example of an `-edit` request for editing records in a portal, where “Address:” is the name of the relationship, “City.1” is the first row in the portal, and “City.2” is the second row in the portal:

```
FMPro?-db=Employees.fp5&-lay=LayoutOne&recid=11&
FirstName=Sam&LastName=Smith&Address::City.1=Seattle
&Address::City.2=New York&-format=reply.htm&-edit
```

The following is an example of another `-edit` request for editing records in a portal, in an HTML form:

```
<FORM ACTION="FMPro" METHOD="POST">
```



```

<INPUT TYPE="HIDDEN" NAME="-db"
VALUE="Employees.fp5">
<INPUT TYPE="HIDDEN" NAME="-lay" VALUE="LayoutOne">
<INPUT TYPE="HIDDEN" NAME="-format" VALUE="reply.htm">
<INPUT TYPE="HIDDEN" NAME="-recid" VALUE="11">
<INPUT TYPE="TEXT" NAME="FirstName" VALUE="Sam">
<INPUT TYPE="TEXT" NAME="LastName" VALUE="Smith">
<INPUT TYPE="TEXT" NAME="Address::City.1"
VALUE="Seattle">
<INPUT TYPE="TEXT" NAME="Address::City.2" VALUE="New
York">
<INPUT TYPE="SUBMIT" NAME="-edit" VALUE="Edit Record">

```

```
</FORM>
```

To display records in a portal on your web page, you use a [FMP-Portal] replacement tag in the format file. For information, see the CDML Reference database (described in “About the CDML Reference database” on page 9-9). For an example of displaying portals in a web page, see the Shopping Cart example (described in “Shopping Cart example” on page 9-15.)

Using the CDML Tool and templates

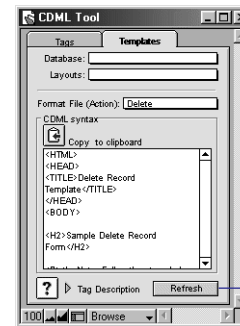
The CDML Tool is a special FileMaker Pro database that you can use with any open database and your HTML authoring program to copy and paste tags into your format files. It also includes HTML templates for the main types of format files and comments for how to use them.

Your databases must be open and shared via the Web Companion. For information, see “Enabling the Web Companion” on page 8-3 and “Sharing the database via the Web” on page 8-5.

When you specify the name of an open database in the CDML Tool, the database’s layout, field, and value list names will automatically appear in pop-up menus for you to choose from. Other database-specific information will appear as appropriate within the CDML tags.

To use the CDML Tool:

1. Open the CDML_Tool.fp5 database located in the CDML folder. Developer Extras\FileMaker, Inc\External FileMaker APIs\CDML\Web Tools\CDML Tool.fp5



The Templates tab in the CDML Tool

2. Open the database that you’re planning to publish on the Web.
3. In a text editor or HTML authoring program, create a blank page.
4. Arrange the windows so you can easily go between the blank page and the CDML Tool.
5. In the CDML Tool, select either the Templates tab or the Tags tab and click the Refresh button. Then choose your database from the Database pop-up menu, and the layout that you want to use from the Layouts pop-up menu.

The layout you choose determines which fields will be accessed. Otherwise, it does not affect the layout of the web page.

Using the Templates tab

1. With the Templates tab selected, choose a format file from the Format File (Action) pop-up menu for the type of action that you want users to perform on your database.

For a description of each format file, see “Customizing a format file template” on page 9-6.

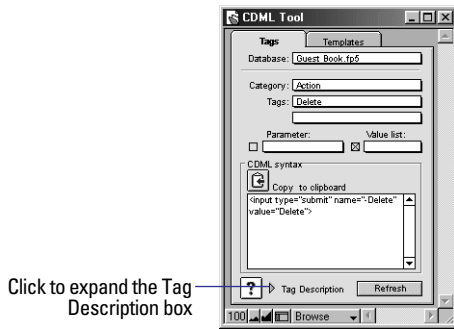
2. Click the Copy to clipboard button.

3. Paste the template into your blank HTML page. (If you're using an HTML authoring program, make sure to paste the template into the HTML source.)

Using the Tags tab

1. With the Tags tab selected, choose a category from the Category pop-up menu and then click the Refresh button to update the type of tags listed in the Tags pop-up menu.

For a description of the categories, see “Categories of CDML tags” on page 9-8.



The Tags tab in the CDML Tool

2. Choose a tag from the Tags pop-up menu.

If the tag has a corresponding list of parameters or value list options, those values will appear in the Parameters and Value List pop-up menus.

3. Choose a parameter or value list option, if applicable, and click the check box on the left to select your choice. When a value list option is selected, the parameter appears in the CDML Syntax box.

For example, the Check boxes tag in the Fields (Add) Dynamic category includes a list of encoding parameters you can choose from (Raw, HTML, Break, Display, and URL) and a choice of two value list parameters (Value List Name or Yes/No).

[FMP-ValueList: Value List Name]

[FMP-ValueList: Yes/No]

The HTML parameter is selected by default for the [FMP-Field] replacement tag.

For information, see “Using an encoding parameter with a CDML replacement tag” on page 9-12 and the individual tag descriptions in the Tag Index tab of the CDML Reference (described in “About the CDML Reference database” on page 9-9).

4. View the tag's syntax in the box at the bottom of the CDML Tool window.

You can also click the Tag Description arrow to expand the window and view a description of the tag in another box below.

5. Click the Copy to clipboard button.

6. Paste the tag into your format file.

Customizing a format file template

The CDML Tool provides nine format file templates for the basic type of actions that web users can perform with your database. These templates are included in the CDML Templates folder on the *FileMaker Developer CD*.

Choose this template

So web users can do this with your database

Delete.htm	Delete a selected record
Delete_reply.htm	Receive feedback after deleting a record
Detail.htm	View one record in detail on the screen

Choose this template	So web users can do this with your database
Edit.htm	Make changes to one record
Edit_reply.htm	Receive feedback after editing a record
New.htm	Add a record
New_reply.htm	Receive feedback after adding a record
Results.htm	View the results of a search for records
Search.htm	Search for specific records

To use a format file template:

1. Copy and paste the template from the CDML Tool into the HTML source of a blank document, or open the template text file from the CDML Templates folder (inside the CDML folder) in a text editor or HTML authoring program.

Developer Extras\FileMaker, Inc\External FileMaker APIs\CDML\CDML Templates

Note When you're familiar with the template, you may want to delete the comments (including the comment tags `<!-- -->`).

```

<HTML>
<HEAD>
<TITLE>Delete Record Template</TITLE>
</HEAD>
<BODY>

<H2>Sample Delete Record Form</H2>

<P><FORM ACTION="FMPpro" METHOD="post">
<P><INPUT TYPE="hidden" NAME="--db" VALUE="Database Name">
<P><INPUT TYPE="hidden" NAME="--lay" VALUE="Layout Name">
<P><INPUT TYPE="hidden" NAME="--format" VALUE="path-to-file/deletereply.htm">
<P><Field Name1<BR>
<INPUT TYPE="text" NAME="Field Name1" VALUE="[FMP-field: Field Name1]">
<P><Field Name2<BR>
<INPUT TYPE="text" NAME="Field Name2" VALUE="[FMP-field: Field Name2]">
<P><INPUT TYPE="hidden" NAME="--RecID" VALUE="[FMP-currentrecid]">
<P><INPUT TYPE="submit" NAME="--delete" VALUE="Delete This Record">
<P><INPUT TYPE="reset" VALUE="Clear Form">
</FORM>
</BODY>
</HTML>
    
```

The delete.htm format file with all `<!--comments-->` removed

2. Use the CDML Tool to add any CDML tags for the fields and other types of data you're requesting from the database.

See "Categories of CDML tags" next.

3. Add text, images and links as desired.

4. Name and save the file in your web site.

This template	Contains these CDML tags
Delete.htm	-db, -lay, -format, [FMP-Field: Field Name1], [FMP-Field: Field Name2], [FMP-CurrentRecID], and -delete
Delete_reply.htm	[FMP-LinkRecID: Layout=Layout Name, Format=path-to-file/default.htm]
Detail.htm	-db, -recid, [FMP-CurrentRecID], -lay, -format, [FMP-Field: Field Name1], [FMP-Field: Field Name2], -edit
Edit.htm	-db, -recid, [FMP-CurrentRecID], -lay, -format, [FMP-Field: Field Name1], [FMP-Field: Field Name2], -edit
Edit_reply.htm	[FMP-LinkRecID: Layout=Layout Name, Format=path-to-file/Detail.htm]
New.htm	-db, -lay, -format, -new
New_reply.htm	[FMP-LinkRecID: Layout=Layout Name, Format=path-to-file/Detail.htm]
Results.htm	[FMP-RangeStart], [FMP-RangeEnd], [FMP-CurrentFoundCount], [FMP-RangeSize], [FMP-LinkPrevious], [FMP-LinkPrevious] [FMP-Record] [FMP-Field: Field Name1], [FMP-Field: Field Name2] [FMP-LinkRecID: Layout=layout_name, Format=path-to-file/detail.htm] [FMP-Record] [FMP-LinkNext], [FMP-LinkNext]
Search.htm	-db, -lay, -format, -op -find

For examples of more complex format files, see the CDML examples included with FileMaker Developer. These examples are described in "Planning your web site" on page 9-12.

Categories of CDML tags

There are three types of FileMaker CDML tags:

- **Action tags** — these tags are used to make a specific request to the database, such as to add a record. Action tags always begin with a hyphen, such as the `-new` tag. (See “Generating FileMaker Pro CGI requests using CDML” on page 9-3 for information.)
- **Variable tags** — these tags are used to specify the parameters of a request, such as the names of the database and the layout. They also begin with a hyphen, such as the `-db` and `-lay` tags.
- **Replacement tags** — these tags are used to display data from the database on a web page. They act as placeholders until a request has been submitted and the requested data is returned to the page. Replacement tags always begin and end with a square bracket, for example, `[FMP-Field: First Name]`.

The CDML Tool organizes the CDML tags and HTML form tags into 15 categories:

Use tags in this category	For this type of interaction with FileMaker Pro
Action	Delete, duplicate, edit, find, or add a record, find all records, find a random record, reset a form, or view a format file containing replacement tags or value lists
Email	Display an email form (containing BCC, CC, Format, From, Host, Subject, To, or All Mail Tags)
Fields (Add) Dynamic	Display dynamic field name and value list information in HTML form elements (text field, text area, check box, radio button, repeating fields, scrolling list) and CDML replacement tags such as <code>[FMP-ValueList]</code> so web users can view and modify current FileMaker Pro data on the web page. When changes are made in the database, the current information is displayed on the web page.

Use tags in this category	For this type of interaction with FileMaker Pro
Fields (Add) Static	Display static field name and value list information from the database in HTML form elements (text field, text area, check box, radio button, repeating fields, scrolling list). This information does not change on the web page regardless of changes made in the database.
Fields (Display)	Display dynamic field name and value list information in HTML form elements (text field, text area, check box, radio button, repeating fields, scrolling list) and display field data in <code>[FMP-Field]</code> , <code>[FMP-ValueListItem]</code> , and <code>[FMP-RepeatingItem]</code> replacement tags. Web users cannot edit this data.
Fields (Update)	Display dynamic field name and value list information in HTML form elements (text field, text area, check box, radio button, repeating fields, scrolling list) and display field data in <code>[FMP-Field]</code> , <code>[FMP-ValueListItem]</code> , <code>[FMP-RepeatingItem]</code> , and <code>[FMP-Option]</code> replacement tags within <code>INPUT</code> and <code>SELECT</code> elements so web users can edit this data
Find Operators	Use an operator (<code>AND/OR</code> , <code>OR</code> , hidden, number/dates, or text) when performing a <code>-find</code> request
Hidden	Do not display this <code>INPUT</code> tag (<code>-db</code> , <code>-lay</code> , <code>-db & -lay</code> , <code>-error</code> , <code>-format</code> , or <code>-token</code>)
Links	Display a format file based on CDML replacement or action tags within <code>HREF</code> , <code>MAILTO</code> , and <code>SRC</code> (image) links
Logical	Conditionally display data within the <code>[FMP-If]</code> and <code>[FMP-Else]</code> replacement tags
Looping	Display multiple lines of data within one of these looping types of replacement tags (Current Find, Current Sort, Layout Fields, Portal, Record, Repeating Fields, Value List, or Value Names)
Names Only	Display a list of database, field, layout, script, or value list names from any open database

Use tags in this category	For this type of interaction with FileMaker Pro
Replacement	Display specific data from the database in one of 44 types of replacement tags on the web page. For example, display the web user's IP address in the [FMP-ClientIP] replacement tag.
Variables (Add)	Generate information from the client (web user's) computer based on one of eight replacement tags received as parameters to a request in the FileMaker Pro CGI command: [FMP-ClientAddress], [FMP-ClientIP], [FMP-ClientType], [FMP-ClientUserName], [FMP-CurrentDate], [FMP-CurrentTime],[FMP-CurrentDay], and [FMP-CurrentToken]
Variables (Display)	Display information in one of 21 replacement tags that correspond to a specific request parameter. For example, display the maximum number of records in the [FMP-MaxRecords] replacement tag as specified by the -max request parameter (CDML variable tag).

Using an intratag parameter

You can add a parameter to certain replacement tags that is based on the contents of fields and other items.

Any first parameter that is allowed with the [FMP-If] tag on the left side of an operator, such as CanDelete, ClientPassword or ValueListItem, can be used as a third parameter on the right side of an operator with these replacement tags, as long as they're within curly brackets { }. (See the [FMP-If] tag syntax in the CDML Reference for a description of the first parameters for the [FMP-If] tag.)

Note Some restrictions apply to using the intratag parameter. See the CDML Reference for a description of each replacement tag and how the intratag parameter may be used.

Modified replacement tags that allow for the intratag parameter include:

[FMP-If: {intratag}]

[FMP-Cookie: {intratag}]

[FMP-InlineAction: {intratag}]

[FMP-Log: {intratag}]

[FMP-SetCookie: {intratag}]

[FMP-ContentType: {intratag}]

[FMP-CurrentToken: {intratag}]

[FMP-LinkRecID: {intratag}]

[FMP-Field: {intratag}]

[FMP-Option: {intratag}]

[FMP-Repeating: {intratag}]

[FMP-ValueList: {intratag}]

About the CDML Reference database

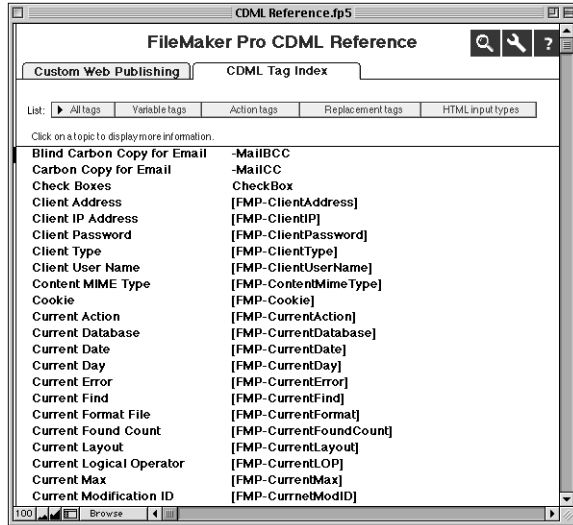
The CDML Reference database is divided into two parts:

- CDML Tag Index — an index for all of the CDML tags with topics that describe what each tag does and how it appears in syntax
- Custom Web Publishing — general guidelines for custom web publishing using CDML

To view the CDML Reference database:

1. Open the CDML Reference.fp5 file:

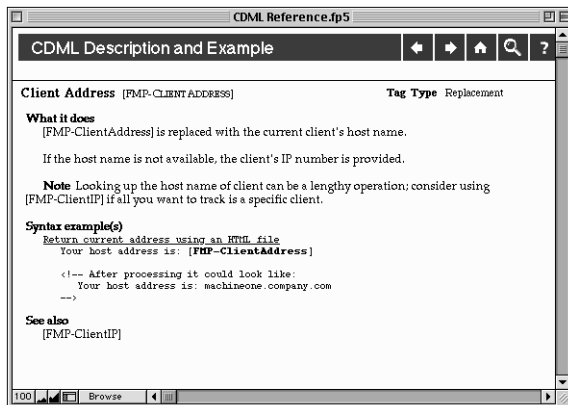
Developer Extras\FileMaker, Inc\External FileMaker APIs\CDML\Web Tools\CDML Reference.fp5



2. On the CDML Tag Index tab, click a button to see a list of tags grouped by one of these categories: All tags, Action tags, Variable tags, Replacement tags, or HTML Input Types (form elements).

3. Then click a tag to display a topic about it.

Each topic explains how the CDML tag is used and its syntax.



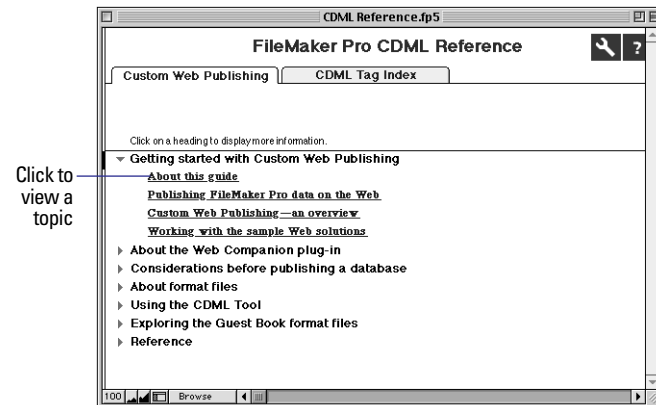
4. Click the home button to return to the FileMaker Pro CDML Reference screen.



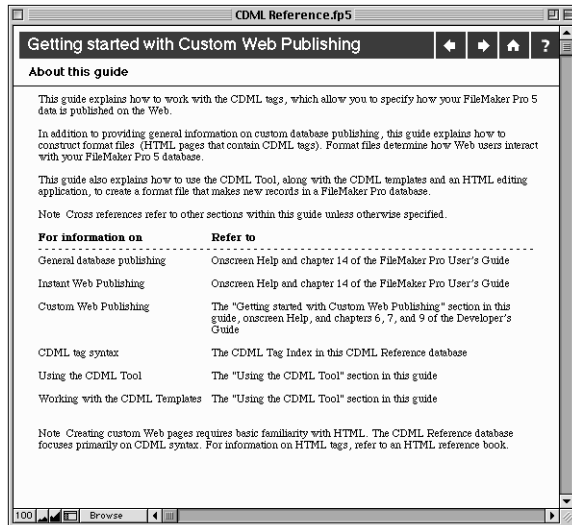
From the home screen, you can open the CDML Tool by clicking the tool button.



5. Click the Custom Web Publishing tab to view general information about custom web publishing using CDML.



6. Click an arrow or a subject title in the table of contents to view a list of underlined topic titles. Then click an underlined title to view the topic.



7. Click an arrow button to continue reading the topics in sequence.

Creating error messages

The FileMaker Pro Web Companion automatically generates an error page for various errors that occur when web users access a database. You can specify your own error pages to override the built-in pages in the following ways:

- Use the `–error` variable tag in a FileMaker Pro CGI request to specify a single error page, which will be displayed for every type of error that occurs.
- Use the `–error` variable tag in the request to specify an error page that contains `[FMP-If]` and `[FMP-Else]` replacement tags and FileMaker Pro error codes, which determine certain conditions for displaying an error message.
- Include `–errnum` along with the `–error` tag in the request to specify a range or single error code. You can use multiple `–errnum` tags to specify discontinuous ranges, such as 500-510 and 900-978.

- Include a specific error page with a filename that the Web Companion recognizes when it's located in a folder named "Error" inside the Web folder.

The `–error` tag in a FileMaker Pro CGI request overrides both the built-in error pages and the pages within the Error folder.

For any error page not included in the Error folder, the Web Companion generates a built-in error page instead.

The Web Companion recognizes these error pages in the Error folder:

NoResults.htm
 ReqFieldMissing.htm
 RepRelatedField.htm
 UnexpectedError.htm
 NotEnoughMemory.htm
 DatabaseNotOpen.htm
 LayoutNotFound.htm
 DataEntryError.htm
 DatabaseViolation.htm
 FieldViolation.htm
 SecurityDisabled.htm
 FormatNotFound.htm
 FileNotFound.htm
 InvalidAddress.htm

For a complete list of error code numbers, see appendix C, "FileMaker Pro values for error codes."

For examples of error pages, see the `gb_error.htm` file in the Guest Book example, the `errors.htm` file in the Employee Database example, and the `reqfielderror.htm` file in the Shopping Cart example. (See “Looking at the three CDML examples” on page 9-13 for more information.)

Developer Extras\FileMaker, Inc\External FileMaker APIs\CDML\CDML Examples

Using an encoding parameter with a CDML replacement tag

You can use a special encoding parameter with certain CDML replacement tags to specify how the data will be encoded by the web browser when it is sent to the web page.

[FMP-Field: Information, Break]

This encoding parameter	Tells the browser to do this
Raw	Don't perform any encoding
HTML	Use standard HTML encoding
Format (new parameter)	Use standard HTML encoding and replace text, numbers, dates, times, and container size formatting with FileMaker Pro formats (for example, to add <BOLD> and <ITALIC> elements) This new parameter can be used with the [FMP-Field] and [FMP-RepeatingItem] replacement tags. (See “Using an intratag parameter” on page 9-9.)
Break	Use standard HTML encoding and replace soft carriage returns with the HTML element

This encoding parameter	Tells the browser to do this
Display	Translate text to the language specified in the Web Companion Configuration dialog box. Use this parameter with the [FMP-CurrentAction], [FMP-FindOpItem], and [FMP-SortOrderItem] replacement tags. For information, see the CDML Reference database (described in “About the CDML Reference database” on page 9-9) and “Setting Web Companion configuration options” on page 8-3.
URL	Use URL encoding, which includes converting spaces to %20.

The encoding parameters are displayed in the **Parameters** pop-up menu in the CDML Tool when the appropriate CDML replacement is selected. (See “Using the Tags tab” on page 9-6.)

Planning your web site

You'll need to create a format file for every type of interaction with FileMaker Pro that you want your web site to provide. As you create each format file, you'll need to know

- what web users will do on the page
- what type of requests will be made to the database
- which format file will be displayed as a result of each request

You'll also need to create pages to display error messages and other types of feedback.

As you add CDML tags to a format file, you'll need to know in advance what the names are of the database, the layout you want to use, and the next format file in the sequence. A flowchart can be useful to map out the page links and interactions with each database.

Three examples are included with FileMaker Developer to give you ideas on how to organize a site. You can also use them as templates and modify them for your own sites.

Looking at the three CDML examples

The *FileMaker Developer* CD includes three CDML examples that demonstrate ways to publish your databases on the Web using CDML. These examples are located in the CDML folder:

Developer Extras\FileMaker, Inc\External FileMaker APIs\CDML\CDML Examples

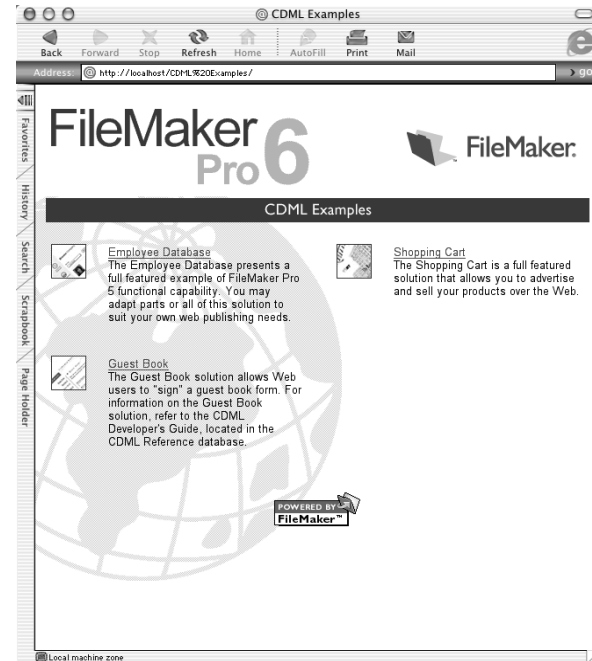
To examine the three CDML examples:

1. Copy the CDML Examples folder and its contents into the root level of the Web folder, located inside the FileMaker Developer application folder. The contents include one default.htm file, an Images folder, and the Employee Database, Guest Book, and Shopping Cart folders.
2. In FileMaker Pro, open each database file inside the three example folders and make sure that each one is shared via the Web Companion. See “Enabling the Web Companion” on page 8-3 and “Sharing the database via the Web” on page 8-5 for information.)
3. In your web browser, type `http://localhost` or your computer’s IP address followed by `/CDML_Examples/` and press Enter.

`http://localhost/CDML_Examples/`

The Web Companion displays the default.htm file located inside the CDML Examples folder.

For information on setting up your computer as a localhost, see “Testing your site without a network connection” on page 8-17.



Click a link to go to the default.htm file for each CDML example

4. Click the links on the Web Companion Demonstration page to go to each of the three examples.
5. As you explore each example site, view the HTML source on each page to see how CDML is used.

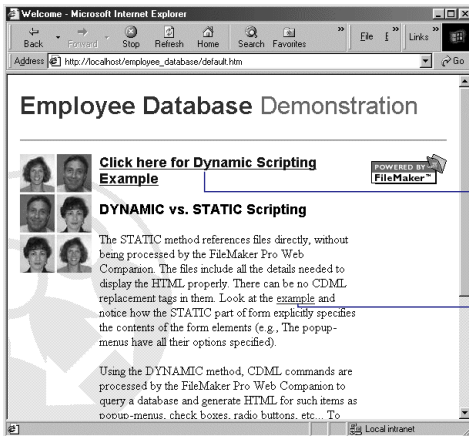
Be sure to examine the FileMaker Pro CGI commands in the HREF links or HTML forms. (See “Generating FileMaker Pro CGI requests using CDML” on page 9-3.)

Note To see the CDML replacement tags, open the example files directly in the browser (without using the web site example links).

Employee Database example

The Employee Database example is designed to demonstrate the most widely used CDML tags. It shows how to

- search for records in the database
- select sorting options
- browse the database
- preselect how many records to view at a time
- add a record to the database



Click to search, browse, and add records to the Employees.fp5 database

Click to display the example.htm page statically

The Employee Database example also provides a demonstration of the difference between using a static HTML form and dynamic CDML to display a value list. Examine the difference between two [example](#) links that both go to the same example.htm page but only one dynamically displays value lists (data) from the database.

The first [example](#) link is a simple HREF link to the example.htm page.

```
<A HREF="example.htm">example</A>
```

And the CDML replacement tags appear statically on the page.

```
Shift: [FMP-valuelist: Shift] ⌘ [FMP-valuelistitem] [/FMP-valuelist]
```

The second [example](#) link contains a FileMaker Pro CGI command for a CDML -view request. (The -view CDML action tag is usually used to display format files containing value lists or CDML replacement tags.)

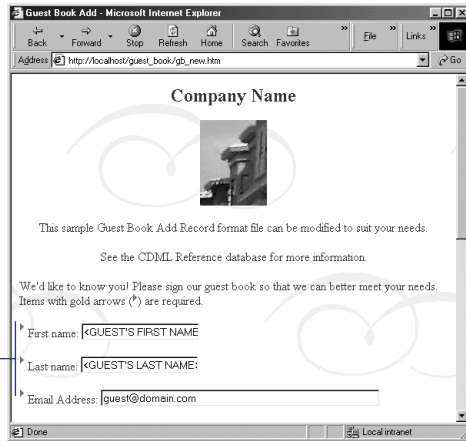
```
<A HREF=FMPPro?-db=Employees.fp5&-lay=Detail&-format=example.htm&-view">example</A>
```

The CDML replacement tags are replaced with the value lists from the database.

```
Shift: ⌘ Part Time ⌘ Full Time
```

Guest Book example

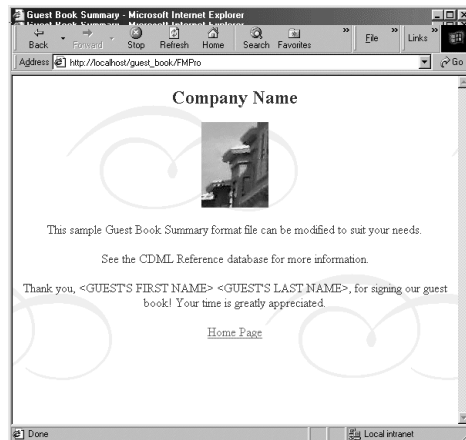
The Guest Book example demonstrates how web users can enter information on a web page (in an HTML form) and submit it to a new record in a FileMaker Pro database. The HTML form that is used on the New Record web page includes CDML tags that allow the database to be accessed and a new record to be created.



Arrows indicate required fields in the database

The New Record page in the Guest Book example

Once a guest signs in and sends the information, a summary page is returned notifying the guest that the information has been received. The summary page contains [FMP-Field] replacement tags for the first and last name of the guest that was entered in the new record.

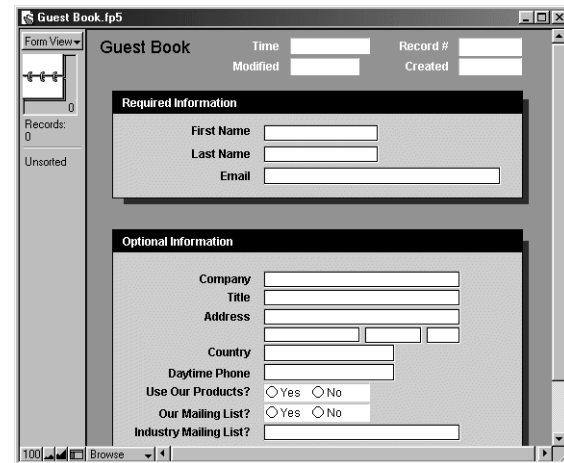


The Guest Book summary page shows the web user's first and last names from the new record

Arrow icons on the New Record page indicate three fields that are required fields in the database. If any of these fields is left blank when the information is submitted to the database, then the error message page (gb_err.htm) is displayed. The error page is specified on the New Record page in a hidden INPUT element.

```
<INPUT TYPE="hidden" NAME="-error" VALUE="gb_err.htm">
```

For information, see “Creating error messages” on page 9-11.

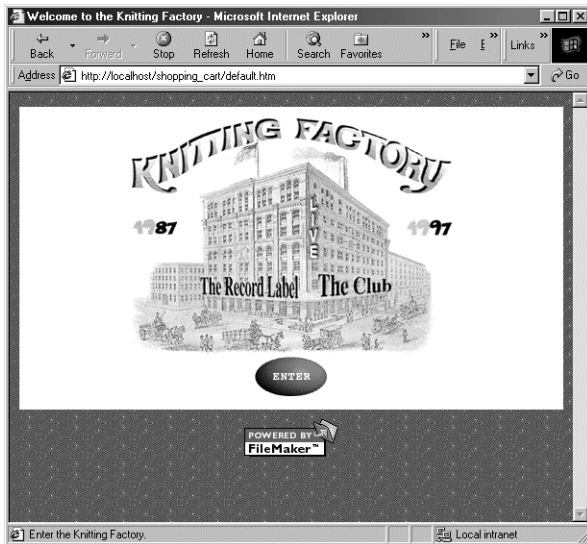


Web users add records to the empty Guest_Book.fp5 database

Shopping Cart example

The Shopping Cart example is a complex web site for the Knitting Factory record label company. The site includes five databases for searching and browsing the company's product list, listening to audio demos, and placing orders.

The site keeps track of each web user by passing CDML tokens from one web page to the next. Some of the CDML format files use the [FMP-Include] replacement tag to reference text files and include their content on the web pages. In addition, the site uses scripting for easy navigation, which is described in the nav.js JavaScript library.



The default.htm page in the Shopping Cart Example folder

Items that are placed in the Shopping Cart are added to the OrderedItems.fp5 database. Information that the web user enters on the customerid.htm page is added to a new record in the Customers.fp5 database. Items that are finally ordered (when the web user proceeds to Checkout and clicks Continue) are added to the Orders.fp5 database. The status of these orders remain open until the web user enters a Knitting Factory account number and expiration date.

All format files in these examples are well-commented, explaining how the CDML tags are used in every step.

Chapter 10

Using FileMaker Pro XML to deliver your data on the Web

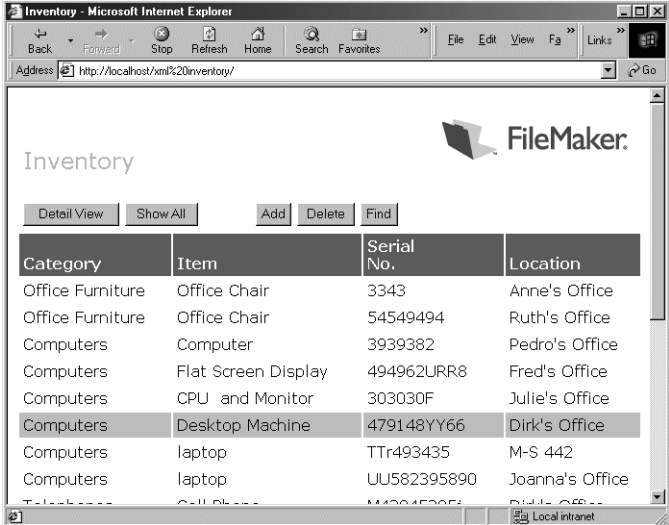
Using the Web Companion, FileMaker Pro can deliver data from your database to the Web in Extensible Markup Language (XML) format. In the same way that HTML has become the standard display language for communication on the World Wide Web, XML promises to become the standard language for structured data interchange.

In XML format, FileMaker Pro data can be populated in your web page programmatically instead of downloading statically from the server. This gives you more flexibility and a more web-like application that allows your web users to interact with the data after it has been downloaded. This also allows the web server to handle more requests as more processing is done by the browser on web users' machines.

Note See the FileMaker Pro *Getting Started Guide* and the FileMaker Pro online Help for information about importing and exporting data in XML format.

About the XML examples

The *FileMaker Developer CD* includes an XML example that demonstrates how you can publish your database on the Web using XML and Dynamic HTML (including JavaScript). This XML example is designed to work in the Microsoft Internet Explorer 5 for Windows web browser. For step-by-step instructions, see “Looking at the XML Inventory example” on page 10-17.



Category	Item	Serial No.	Location
Office Furniture	Office Chair	3343	Anne's Office
Office Furniture	Office Chair	54549494	Ruth's Office
Computers	Computer	3939382	Pedro's Office
Computers	Flat Screen Display	494962URR8	Fred's Office
Computers	CPU and Monitor	303030F	Julie's Office
Computers	Desktop Machine	479148YY66	Dirk's Office
Computers	laptop	TTr493435	M-S 442
Computers	laptop	UU582395890	Joanna's Office
Telephones	Cell Phone	44304330F	Julie's Office

List View of the Inventory.fp5 database in the XML Inventory example

For examples showing the differences between using stylesheets or scripting (Dynamic HTML) with your FileMaker Pro XML documents, see “Comparing CSS, XSLT, and JavaScript” on page 10-11.

For general information on XML (including a glossary of XML terms), additional examples that use XML, and links to XML resources, see the FileMaker, Inc. web site at www.filemaker.com. As a shortcut to the site, in FileMaker Pro, choose Help menu > FileMaker on the Web.

General process for custom web publishing using XML

Here's a simple overview of the process for publishing your FileMaker Pro database on the Internet or an intranet using XML:

1. You send a FileMaker Pro CGI request (such as to find records in the database) to the Web Companion through an HTML form, an HREF link, or a script on your web page. The request can also be made by typing the URL in the web browser.

See "Generating FileMaker Pro CGI requests for an XML document" on page 10-8 and appendix B, "Valid names used in CGI requests for FileMaker Pro XML data."

2. The Web Companion generates an XML document containing the results of your request in XML format (for example, a found set of records from the database and an XML-style sheet processing instruction) and returns it to your web browser.

See "Generating an XML document" next.

3. The web browser, with the help of an XML parser, applies any instructions that you've specified via a style sheet and displays the data in HTML format.

See "Using style sheets with your XML document" on page 10-10.

Once the XML document is downloaded to your web browser, you can use style sheets (such as CSS or XSL) to apply text formatting styles and object positioning, or scripting (such as JavaScript) to manipulate the data however you want. See "Comparing CSS, XSLT, and JavaScript" on page 10-11.

Generating an XML document

When you specify an XML format parameter in your FileMaker Pro CGI request, the Web Companion generates an XML document containing data from your database that is formatted by one of two types of XML grammars (or schemas).

One type (called FMPDSO) gives you more flexibility and control over individual elements and is ideally suited for use with cascading style sheets (CSS) or Extensible Stylesheet Language (XSL). The FMPDSO grammar can also be used with the Microsoft XML Data Source Object (DSO) in Internet Explorer 4.0 to publish read-only databases. (The Microsoft XML DSO lets you view but not update data in XML format.)

The other type of grammar (called FileMaker Extended XML or FMPXML) provides a broader, richer XML that defines FileMaker Pro layouts, fields, and value list information. These grammars can be combined with XSL documents or scripting (such as JavaScript) to publish dynamic databases on the Web.

All XML data generated by the Web Companion is well-formed and compliant with the XML 1.0 specification. The document type definitions (DTDs) for the grammars are provided in HTML documents for your convenience (included on the *FileMaker Developer* CD).

Developer Extras\FileMaker, Inc\External FileMaker APIs\XML\Documentation

Two of the grammars generated by the Web Companion are used for retrieving query results and a third is used for retrieving layout information. Depending on what you specify in your FileMaker Pro CGI request, the Web Companion will generate an XML document using one of these grammars:

- the FMPDSORESULT grammar
- the FMPXMLRESULT grammar
- the FMPXMLLAYOUT grammar

Each XML document contains a default XML namespace declaration for the grammar. (See "About XML namespaces" next.) You can also specify that the document contain an XML-style sheet processing instruction. (See "Using style sheets with your XML document" on page 10-10.)

Use one of these grammars in your document or web page to display and work with FileMaker data in XML format.

Note XML data generated by the Web Companion is encoded using UTF-8 format (Unicode Transformation Format 8). For information, see “About UTF-8 encoded data” on page 10-8.

About XML namespaces

To avoid name collisions, unique XML namespaces help distinguish XML tags by the application they were designed for. For example, if your XML document contains two DATABASE elements, one for FileMaker Pro XML data and another for Oracle XML data, the namespaces will identify the DATABASE element for each.

The FileMaker Pro Web Companion generates a default namespace for each grammar. For example, for the FMPDSORESULT grammar, the following namespace is generated:

```
xmlns="http://www.filemaker.com/fmpdsoreult"
```

About FileMaker Pro database error codes

The FileMaker Pro Web Companion generates an error code at the beginning of each grammar based on the current error status of the database. A value of zero (0) is returned for no error.

```
<ERRORCODE>0</ERRORCODE>
```

See appendix C, “FileMaker Pro values for error codes.” for information.

Using the FMPDSORESULT grammar

When you specify “-dso_xml” as the format for a FileMaker Pro CGI request, the Web Companion will generate XML data based on a database-specific grammar that uses field names as element names. The FMPDSORESULT grammar is useful for publishing databases on web pages that are formatted with cascading style sheets or XSLT. (See “Comparing CSS, XSLT, and JavaScript” on page 10-11 for information.) The FMPDSORESULT grammar is compatible with the Microsoft XML Data Source Object (DSO) used by Internet Explorer, and is available on the *FileMaker Developer CD* (Developer Extra\fileMaker, Inc\External FileMaker APIs\XML\Documentation\fmxmlresult_dtd.htm).

The Web Companion will also generate the document type definition for the grammar if you specify “-dso_xml_dtd” as the format. This is useful if you want an XML parser to validate the XML before your document goes to production.

Note Beginning with Internet Explorer 4.0, Internet Explorer has directly supported XML without the requirement of additional software. The XML can be displayed using dynamic data binding features available in the browser. This is accomplished with a Java applet that ships with Internet Explorer, which presents the XML as a DSO to the browser. With the DSO, the Internet Explorer browser exposes XML data to scripting languages such as JavaScript or VBScript via the Microsoft Document Object Model (DOM). Keep in mind that the Microsoft XML DSO applet does not provide a mechanism for updating the data, nor does it know anything about FileMaker Pro database layouts or value lists. In addition, future browser updates could change, limit, or remove the functionality described above.

The following is an example of a Microsoft XML DSO applet tag that you might use in your web page to query FileMaker Pro for XML data using the FMPDSORESULT grammar—where the “url” parameter can be any valid FileMaker Pro CGI request containing a –format parameter equal to “–dso_xml” or “–dso_xml_dtd.” (See “Generating FileMaker Pro CGI requests for an XML document” on page 10-8 for a list of valid FileMaker CGI requests.)

```
<applet code=com.ms.xml.dso.XMLDSO.class width=0 height=0
id=xmldso MAYSCRIPT=true>
  <PARAM NAME="url" VALUE=fmpro?-db=PhoneList.fp5&
  -format=-dso_xml&-find=>
</applet>
```

Description of elements in the FMPDSORESULT grammar

Each ROW element in the generated FMPDSORESULT grammar contains a number of FIELD elements that correspond to the field names in the specified layout.

Spaces or single colons in field names are converted to underscores in the element names (for example, <FIRST_NAME>). Double colons in portal fields are converted to periods (for example, <PHONE.PHONE_NUMBER>). This is done because colons are reserved in XML for specifying namespaces and spaces are not allowed in XML element names.

For repeating and portal fields, each FIELD element will contain a DATA element that corresponds to each repetition or portal record.

Note The content of container fields in the database will be generated in the form of the relative URL used for retrieving the content instead of the actual content (such as an image).

To qualify the XML elements for the FileMaker Pro application, the names of all elements and attributes in this grammar are associated with the unique XML namespace `http://www.filemaker.com/fmpdsoreresult`. This namespace is declared in the grammar as the default namespace.

The following is an example of XML data generated with the FMPDSORESULT grammar.

Example of XML data in the FMPDSORESULT grammar

```
<?xml version="1.0" encoding="UTF-8"?>
<FMPDSORESULT xmlns="http://www.filemaker.com/fmpdsoreresult">
  <ERRORCODE>0</ERRORCODE>
  <DATABASE>PhoneList.fp5</DATABASE>
  <LAYOUT>Web Layout</LAYOUT>
  <ROW RECORDID="3" MODID="23">
    <FIRST_NAME>John</FIRST_NAME>
    <LAST_NAME>Smith</LAST_NAME>
    <PHONE.PHONE_NUMBER>
      <DATA>555-444-3333</DATA>
      <DATA>555-222-9999</DATA>
    </PHONE.PHONE_NUMBER>
  </ROW>
  <ROW RECORDID="6" MODID="32">
    <FIRST_NAME>Barbara</FIRST_NAME>
    <LAST_NAME>Jones</LAST_NAME>
    <PHONE.PHONE_NUMBER>
      <DATA>555-666-7777</DATA>
      <DATA>555-333-0000</DATA>
      <DATA>555-111-7654</DATA>
    </PHONE.PHONE_NUMBER>
  </ROW>
</FMPDSORESULT>
```


Note If the `-lay` parameter is not specified in the FileMaker Pro CGI request, the `LAYOUT` element is empty and data for every field in the database is returned. (See “Generating FileMaker Pro CGI requests for an XML document” on page 10-8 for information.)

Using the FileMaker Pro Extended XML grammars

The FileMaker Pro Extended XML grammars contain additional information about field types, value lists and layouts that is not found in the `FMPDSORESLT` grammar. Use the `FMPXMLRESULT` and `FMPXMLLAYOUT` grammars if you require layout information or want the `METADATA` information provided by these grammars. The `FMPXMLRESULT` and `FMPXMLLAYOUT` grammars are available on the *FileMaker Developer* CD (Developer Extras\FileMaker, Inc\External FileMaker APIs\XML\Documentation)

Note These grammars are not well suited for cascading style sheets with positioning. See “Using the `FMPDSORESLT` grammar” on page 10-3 if you want to use CSS with your XML data.

When you specify “`-fmp_xml`” as the format for a FileMaker Pro CGI request, the Web Companion will generate XML data using either the `FMPXMLRESULT` or `FMPXMLLAYOUT` grammar, depending on the request you specify in the CGI command:

- The Web Companion will generate the `FMPXMLRESULT` grammar when you specify `-edit`, `-delete`, `-find`, `-new`, `-dbnames`, `-layoutnames`, `-scriptnames` or `-dbopen` as the FileMaker CGI request.
- The Web Companion will generate the `FMPXMLLAYOUT` grammar when you specify `-view` as the FileMaker CGI request.

The Web Companion will also generate the document type definition for the grammar if you specify “`-fmp_xml_dtd`” as the format. This is useful if you want an XML parser to validate the XML before your document goes to production.

For a list of valid FileMaker CGI requests, see “Generating FileMaker Pro CGI requests for an XML document” on page 10-8.

Note When using XML grammars, you should do a case-insensitive compare for proper results.

Description of elements in the `FMPXMLRESULT` grammar

In the generated `FMPXMLRESULT` grammar, the `DATABASE` element contains attributes for the name of the database, the number of records in the database, the name of the layout that was used to generate the result set, and the format of dates and times in the XML document.

The `DATEFORMAT` attribute of the `DATABASE` element specifies the format of dates in the XML document.

Field	Full form	Short form
Year	yyyy (4 digits)	yy (2 digits)
Month	mm (2 digits)	M (1 or 2 digits)
Day	dd (2 digits)	d (1 or 2 digits)

The `TIMEFORMAT` attribute of the `DATABASE` element specifies the format of times in the XML document.

Field	Full form	Short form
Hour (1 – 12)	hh (2 digits)	h (1 or 2 digits)
Hour (1 – 24)	kk (2 digits)	k (1 or 2 digits)
Minute	mm	
Second	ss	
AM/PM	a	

The METADATA element of the FMPXMLRESULT grammar contains one or more FIELD elements, each containing information for one of the fields/columns of the result set—including the name of the field as defined in the database, the field type, the Yes or No allowance for empty fields (EMPTYOK attribute) and the maximum number of repeating values (MAXREPEAT attribute). Valid values for field types are TEXT, NUMBER, DATE, TIME, and CONTAINER.

The RESULTSET element of the FMPXMLRESULT grammar contains all of the ROW elements returned as the result of a query and an attribute for the total number of records found. Each ROW element contains the field/column data for one row in the result set—including the record ID for the row, the modification ID for the row, and the COL element containing the data for one field/column in the row (where multiple DATA elements represent one of the values in a repeating or portal field).

Note The content of container fields in the database will be generated in the form of the relative URL used for retrieving the content, instead of the actual content (such as an image).

To qualify the XML elements for the FileMaker Pro application, the names of all elements and attributes in this grammar are associated with the unique XML namespace <http://www.filemaker.com/fmpxmlresult>. This namespace is declared in the grammar as the default namespace.

The following is an example of XML data generated with the FMPXMLRESULT grammar.

Example of XML data in the FMPXMLRESULT grammar

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="yourstylesheet.xsl"?>
<FMPXMLRESULT xmlns="http://www.filemaker.com/fmpxmlresult">
  <ERRORCODE>0</ERRORCODE>
```

```
<PRODUCT NAME="Web Companion" VERSION="6v1"
BUILD="01/24/02"/>
<DATABASE NAME="Employees.fp5" RECORDS="23"
DATEFORMAT="MM/dd/yy" TIMEFORMAT="hh:mm:ss"
LAYOUT="summary"/>
<METADATA>
  <FIELD NAME="First Name" TYPE="TEXT"
EMPTYOK="NO" MAXREPEAT="1"/>
  <FIELD NAME="Last Name" TYPE="TEXT"
EMPTYOK="NO" MAXREPEAT="1"/>
  <FIELD NAME="Department" TYPE="TEXT"
EMPTYOK="YES" MAXREPEAT="1"/>
</METADATA>
<RESULTSET FOUND="5">
  <ROW RECORDID="34" MODID="47">
    <COL>
      <DATA>Joe</DATA>
    </COL>
    <COL>
      <DATA>Smith</DATA>
    </COL>
    <COL>
      <DATA>Engineering</DATA>
    </COL>
  </ROW>
  <ROW RECORDID="78" MODID="89">
    <COL>
      <DATA>Susan</DATA>
    </COL>
    <COL>
      <DATA>Jones</DATA>
    </COL>
```

```

    <COL>
      <DATA>Marketing</DATA>
    </COL>
  </ROW>
</RESULTSET>
</FMPXMLRESULT>

```

The order of the COL elements corresponds with the order of the FIELD elements in the METADATA element—for example, where the “First Name”, “Last Name”, and then “Department” elements are listed in the METADATA, “Joe”, “Smith”, and then “Engineering” are listed in the same order in the RESULTSET ROW.

Note If the `-lay` parameter is not specified in the FileMaker Pro CGI request, the LAYOUT attribute in the DATABASE element is empty and data for every field in the database is returned. (See “Generating FileMaker Pro CGI requests for an XML document” on page 10-8 for information.)

Description of elements in the FMPXMLLAYOUT grammar

In the generated FMPXMLLAYOUT grammar, the LAYOUT element contains the name of the layout, the name of the database, and FIELD elements for each field found in the corresponding layout in the database. Each FIELD element describes the style type of the field, and contains the VALUELIST attribute for any associated value list of the field.

The VALUELISTS element contains one or more VALUELIST elements for each value list found in the layout—each including the name of the value list and a VALUE element for each value in the list.

To qualify the XML elements for the FileMaker Pro application, the names of all elements and attributes in this grammar are associated with the unique XML namespace <http://www.filemaker.com/fmpxmllayout>. This namespace is declared in the grammar as the default namespace.

The following is an example of XML data generated with the FMPXMLLAYOUT grammar.

Example of XML data in the FMPXMLLAYOUT grammar

```

<?xml version="1.0" encoding="UTF-8"?>
<FMPXMLLAYOUT xmlns="http://www.filemaker.com/fmpxmllayout">
  <ERRORCODE>0</ERRORCODE>
  <PRODUCT NAME="Web Companion" VERSION="6v1"
    BUILD="01/24/02"/>
  <LAYOUT NAME="Web Layout" DATABASE="employees.fp5">
    <FIELD NAME="First Name">
      <STYLE TYPE="EDITTEXT" VALUELIST="" />
    </FIELD>
    <FIELD NAME="Last Name">
      <STYLE TYPE="EDITTEXT" VALUELIST="" />
    </FIELD>
    <FIELD NAME="Department">
      <STYLE TYPE="POPUPMENU"
        VALUELIST="Departments" />
    </FIELD>
  </LAYOUT>
  <VALUELISTS>
    <VALUELIST NAME="Departments">
      <VALUE>Engineering</VALUE>
      <VALUE>Marketing</VALUE>
    </VALUELIST>
  </VALUELISTS>
</FMPXMLLAYOUT>

```

About UTF-8 encoded data

All XML data generated by the Web Companion is encoded in UTF-8 (Unicode Transformation 8 Bit) format. This format compresses data from the standard Unicode format of 16 bits to 8 bits for ASCII characters. XML parsers are required to support Unicode and UTF-8 encoding.

UTF-8 encoding includes direct representations of most of the characters used in English using values of 0-127 for the standard ASCII set of characters, and provides multibyte encodings for Unicode characters with higher values. UTF-8 encoded data is compressed almost in half (lower ASCII characters are compressed from 2 bytes to 1 byte), which helps data download faster.

Note Because your XML data is UTF-8 encoded, some upper ASCII characters will be represented by 2 or 3 characters in the text editor—they will appear as single characters only in the XML parser or browser.

The UTF-8 encoding format includes the following features:

- All ASCII characters are one-byte UTF-8 characters. A legal ASCII string is a legal UTF-8 string.
- Any non-ASCII character (i.e., any character with the high-order bit set) is part of a multibyte character.
- The first byte of any UTF-8 character indicates the number of additional bytes in the character.
- The first byte of a multibyte character is easily distinguished from the subsequent bytes. Thus, it is easy to locate the start of a character from an arbitrary position in a data stream.
- It is easy to convert between UTF-8 and Unicode.
- The UTF-8 encoding is relatively compact. For text with a large percentage of ASCII characters, it is more compact than Unicode. In the worst case, a UTF-8 string is only 50% larger than the corresponding Unicode string.

Generating FileMaker Pro CGI requests for an XML document

You use FileMaker Pro CGI (Common Gateway Interface) commands to generate requests for XML data from your database.

For example, to generate a `-find` request to display all employees from a database, web users might click on a link containing the following FileMaker Pro CGI command:

```
FMPPro?-db=employees.fp5&-format=-dso_xml&-styletype=text/css&-stylehref=stylesheet.css&-find
```

Request and parameter names

The following tables list the request and parameter names in name/value pairs you can use in a FileMaker Pro CGI command when requesting data in XML format.

For more information and examples, see appendix B, “Valid names used in CGI requests for FileMaker Pro XML data.”

Use this request name	To generate this request
<code>-new</code>	New record
<code>-edit</code>	Edit record
<code>-delete</code>	Delete record
<code>-find</code>	Find record(s)
<code>-findall</code>	Find all records
<code>-findany</code>	find a random record
<code>-view</code>	View layout info (in FMPXMLLAYOUT grammar)
<code>-dbnames</code>	Retrieve names of all open and web-shared databases
<code>-layoutnames</code>	Retrieve names of all available layouts for a specified open, web-shared database

Use this request name	To generate this request
–scriptnames	Retrieve names of all available scripts for a specified open, web-shared database
–dbopen	Open a database that’s in the Web folder with Remote Administration enabled
–dbclose	Close a database that’s in the Web folder with Remote Administration enabled

Use these parameter names	To go with these requests
–db (database name)	Required for all requests except –dbnames
–lay (layout name)	Required for –view, and with –edit or –new requests for data in related fields and portals. Optional for –find, –findall
–format	Required for all requests. (Use one of these formats: –dso_xml, –dso_xml_dtd, –fmp_xml, or –fmp_xml_dtd)
–recid (record I.D.)	Required for –edit and –delete. Optional for –find
–modid (modification I.D.)	Optional for –edit
–lop (logical operator)	Optional for –find
–op (operator)	Optional for –find
–max (maximum records)	Optional for –find
–skip (skip records)	Optional for –find
–sortorder (sort order)	Optional for –find, –findall
–sortfield (sort field)	Optional for –find, –findall
–script (perform script)	Optional for –find, –findall
–script.prefind (perform script before –find)	Optional for –find, –findall
–script.presort (perform script before sort)	Optional for –find, –findall
–styletype (stylesheet type)	Optional for all requests

Use these parameter names	To go with these requests
–stylehref (stylesheet HREF)	Optional for all requests
–password	Optional for –dbopen requests. Specifies the database’s password.
field name (no hyphen)	At least one field name is required for –new and –edit. Optional for –find. See “field name (Name of specific field)” on page B-10 for more information.

Note The –max parameter now returns 0 if the request returns no records.

Requests for adding records to a portal

When you make an –edit request or a –new request that includes data for a portal of related database records, you must specify the layout and the relationship name for the related database.

Note You can only add one record at a time to a portal, and therefore must make separate –new requests to add more rows to the portal.

The following is an example of a –new request for adding a record to a portal, where “Address:” is the name of the database relationship, and “City.0” is the related field name in the portal:

```
FMPPro?–db=employees.fp5&–lay=LayoutOne&FirstName=Sam
&LastName=Smith&Address::City.0=Seattle&–format= –fmp_xml&–new
```

Requests for editing multiple records in a portal

You only need to make one –edit request to edit multiple records in a portal. You specify each row (or record) in the portal by adding a period and a consecutive number (starting with number 1) to the end of the related field name.

The following is an example of an `-edit` request for editing records in a portal, where "Address::" is the name of the relationship, "City.1" is the first row in the portal, and "City.2" is the second row in the portal:

```
FMPPro?-db=employees.fp5&-lay=LayoutOne&recid=11&
FirstName=Sam&LastName=Smith&Address::City.1=Seattle
&Address::City.2=New York&-format=-fmp_xml&-edit
```

The following is an example of another `-edit` request for editing records in a portal, in an HTML form:

```
<FORM ACTION="fmp" METHOD="POST">
  <INPUT TYPE="HIDDEN" NAME="-db" VALUE="employees.fp5">
  <INPUT TYPE="HIDDEN" NAME="-lay" VALUE="LayoutOne">
  <INPUT TYPE="HIDDEN" NAME="-format" VALUE="-fmp_xml">
  <INPUT TYPE="HIDDEN" NAME="-recid" VALUE="11">
  <INPUT TYPE="TEXT" NAME="FirstName" VALUE="Joe">
  <INPUT TYPE="TEXT" NAME="LastName" VALUE="Smith">
  <INPUT TYPE="TEXT" NAME="Address::City.1" VALUE="San
  Jose">
  <INPUT TYPE="TEXT" NAME="Address::City.2" VALUE="Santa
  Clara">
  <INPUT TYPE="SUBMIT" NAME="-edit" VALUE="Edit Record">
</FORM>
```

Using style sheets with your XML document

The Web Companion will generate an XML-stylesheet processing instruction with each grammar if the FileMaker CGI request includes the `-styletype` and `-stylehref` parameters. This allows you to use cascading style sheets (CSS) or Extensible Stylesheet Language (XSL) documents for displaying your XML document.

The `-styletype` parameter is used for setting the value of the type attribute (`type=text/css` or `type=text/xml`).

The `-stylehref` parameter is used for setting the value of the HREF attribute (`href=document.css` or `href=document.xml`).

Here is an example of what a possible FileMaker CGI command might look like:

```
http://localhost/fmp?db=employees.fp5&-format=-fmp_xml&-find=&
-styletype=text/xml&-stylehref=document.xml
```

Based on this command, the Web Companion will include the following processing instruction in the XML document:

```
<?xml-stylesheet type="text/xml" href="document.xml"?>
```

The following text is an example of a possible XSL document used with the FMPXMLRESULT grammar. In this example, the XSL document converts the XML document into an HTML document by inserting HTML tags. It builds an HTML table that contains a header row for all the field names from the METADATA element in the FMPXMLRESULT grammar, and table rows for all the field data in the ROW elements of the RESULTSET.

Note This is an example of XSLT that was written to work with Internet Explorer 5.0 for Windows, not with other browsers using later versions of XSLT.

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl"
xmlns:HTML="http://www.w3.org/Profiles/XHTML-transitional">
  <xsl:template>
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="text()">
    <xsl:value-of/>
```

```
</xsl:template>
```

```
<xsl:template match="/">
```

```
  <HTML>
```

```
    <BODY>
```

```
      <CENTER>
```

```
        <TABLE BORDER="0">
```

```
          <xsl:apply-templates/>
```

```
        </TABLE>
```

```
      </CENTER>
```

```
    </BODY>
```

```
  </HTML>
```

```
</xsl:template>
```

```
<xsl:template match="ERRORCODE">
```

```
</xsl:template>
```

```
<xsl:template match="METADATA">
```

```
  <TR>
```

```
    <xsl:apply-templates/>
```

```
  </TR>
```

```
</xsl:template>
```

```
<xsl:template match="FIELD">
```

```
  <TD ALIGN="CENTER" BGCOLOR="#336666">
```

```
    <FONT FACE="Verdana"
```

```
      COLOR="#FFFFFF"><B><xsl:value-of  
      select="@NAME"/></B></FONT>
```

```
  </TD>
```

```
</xsl:template>
```

```
<xsl:template match="RESULTSET/ROW">
```

```
  <TR>
```

```
    <xsl:apply-templates/>
```

```
  </TR>
```

```
</xsl:template>
```

```
<xsl:template match="COL">
```

```
  <TD BGCOLOR="#00CCFF">
```

```
    <xsl:apply-templates/>
```

```
  </TD>
```

```
</xsl:template>
```

```
<xsl:template match="DATA">
```

```
  <P><xsl:value-of/></P>
```

```
</xsl:template>
```

```
</xsl:stylesheet>
```

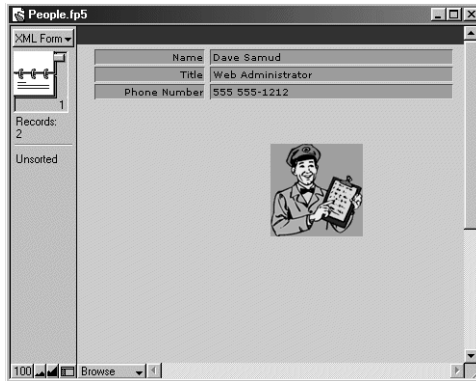
Comparing CSS, XSLT, and JavaScript

The *FileMaker Developer* CD includes three simple examples that demonstrate the differences between using cascading style sheets (CSS), Extensible Stylesheet Language–Transformations (XSLT), and JavaScript scripting language with your XML documents.

These examples are included in the XML folder:

Developer Extras\FileMaker, Inc\External FileMaker APIs\XML

All three examples use a simple database named People.fp5.



The People.fp5 database contains seven fields — three text fields for data, three global fields for field labels, and one container field for pictures.



The seven fields in the People.fp5 database used for these examples

These three examples were designed to be viewed in the Internet Explorer 5.0 for Windows web browser. For information on new browsers that can be used to view the examples, double-click **FileMaker on Web** (included on the *FileMaker Developer CD*) to go to the FileMaker Product support pages in your browser.

To view the examples:

1. Place a copy of the Simple Examples folder and its files in the Web folder inside the FileMaker Developer application folder on your hard disk.

FileMaker Developer \Web \Simple Examples

Note For security, when actually publishing a database on the Web, you should not keep the database file in the Web folder unless you plan to administer it remotely. See “Opening password-protected databases remotely” on page 8-18 for more information.

2. In FileMaker Pro, open the People.fp5 database and make sure it's shared via the Web Companion.

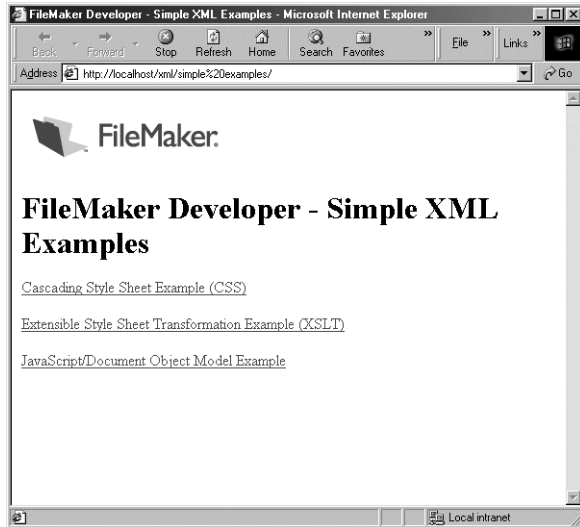
See “Enabling the Web Companion” on page 8-3 and “Sharing the database via the Web” on page 8-5 for information.

3. In your web browser, type localhost (or your computer's IP address) followed by /simple examples/ and press Enter.

<http://localhost/simple examples/>

<http://17.17.17.17/simple examples/>

For information on setting up your computer as the localhost, see “Testing your site without a network connection” on page 8-17.



4. On the Default.htm page, click the links to the CSS, XSL, and Scripting/DOM examples, or view the source to see the FileMaker Pro CGI requests for the links.

See “Generating FileMaker Pro CGI requests for an XML document” on page 10-8 for information.

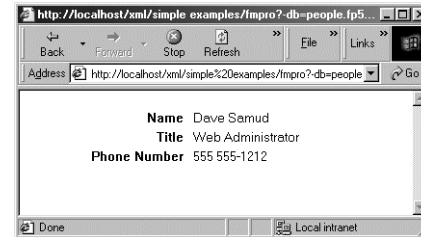
Cascading style sheets (CSS) example

Cascading style sheets (CSS) documents format the text style (font, size, color, etc.) and positioning of your data in an XML document.

A cascading style sheet can only be applied to the existing data in an XML document—it cannot be used to transform the data (such as transforming a URL for a container field into its corresponding image), or to add additional information (such as labels for each field) to the XML document.

This example demonstrates the use of a CSS document with an XML document. The following CGI command was used to generate the FMPDSORESULT grammar for the People.fp5 database and to apply the People_form.css stylesheet to the generated XML data:

```
fmpro?-db=people.fp5&-lay=xml form&-format=dso_xml&-styletype=text/css&-stylehref=people_form.css&-max=1&-find=
```



To simulate field labels, the CSS document applies boldface and right-alignment to data in the three global fields

The picture in the container field is not displayed in this example because the CSS document can only apply styles to the existing data (the relative URL to the image). It’s not possible to transform the URL into its corresponding image using cascading style sheets.

The following text is the people_form.css stylesheet, which adds formatting and positioning to the XML:

```
/*
 * Since FMPDSORESULT is the root element of the XML document,
 * any formatting applied to this element will cascade down to all
 * other elements in the document. This is a good place to set the
 * default formatting options for your page.
 */
FMPDSORESULT
{
    font-family: sans-serif;
    font-size: 10pt;
}
/*
 * Don't display the data for these elements.
 */
```

```

ERRORCODE, DATABASE, LAYOUT, Picture
{
    visibility: hidden;
}
/*
 * Since it is not possible to add additional text to the output
 * of the XML document via CSS, we must rely on the XML document to
 * provide the labels when displaying the field data. In this case,
 * the labels are from global fields defined in the database.
 */
Name_Label
{
    position: absolute;
    left: 20px;
    top: 20px;
    width: 150px;
    text-align: right;
    font-weight: bold;
}
Title_Label
{
    position: absolute;
    left: 20px;
    top: 40px;
    width: 150px;
    text-align: right;
    font-weight: bold;
}
Phone_Label
{
    position: absolute;
    left: 20px;
    top: 60px;
    width: 150px;
    text-align: right;
    font-weight: bold;
}
/*
 * The following styles are applied to the actual field data.
 */
Name
{
    position: absolute;
    left: 180px;
    top: 20px;

```

```

}
Title
{
    position: absolute;
    left: 180px;
    top: 40px;
}
Phone
{
    position: absolute;
    left: 180px;
    top: 60px;
}

```

Extensible Stylesheet Language–Transformations (XSLT) example

Extensible Stylesheet Language–Transformations (XSLT) is a language for transforming XML documents into other XML documents. XSLT is part of the overall XSL specification, which also includes an XML vocabulary for formatting the transformed XML document (such as applying text styles).

This example demonstrates the use of an XSL document with an XML document. The following CGI command was used to generate the FMPDSORESULT grammar for the People.fp5 database and to apply the People_form.xml stylesheet to the generated XML data:

```

fmpro?-db=people.fp5&-lay=xml form&-format=dso_xml&
-styletype=text/xsl&-stylehref=people_form.xml&-max=1&-find=

```



XSL lets you display images in container fields, format the data, and add field labels and a logo to the XML document

The three global fields for field labels in the People.fp5 database are not necessary when you're using an XSL document. You can use XSLT to add labels after the XML document has been generated by FileMaker Pro.

Note You can also include scripting (such as JavaScript) in your XSL document. See “JavaScript scripting language example” next.

The following text is the people_form.xsl stylesheet, which adds labels and transforms the URL in the Picture field into an image:

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl"
xmlns:fm="http://www.filemaker.com/fmpdsoreult">
  <!--
  The following two templates have been added for IE5 Windows,
  since that browser's XSL processor does not provide defaults for
  them.
  -->
  <xsl:template><xsl:apply-templates/></xsl:template>
  <xsl:template match="text()"><xsl:value-of select="."/;></
xsl:template
  <xsl:template match="fm:FMPDSOREULT">
    <html>
```

```
<head>
  <style type="text/css">
    table {font-family: sans-serif; font-size:
      10pt; }
    td.label { text-align: right; vertical-align:
      text-top; font-weight: bold; }
  </style>
</head>
<body>
  
  <xsl:apply-templates/>
</body>
</html>
</xsl:template>
<xsl:template match="fm:ROW">
  <table>
    <tr>
      <td class="label">Name</td>
      <td><xsl:value-of select="fm:Name"/></td>
    </tr>
    <tr>
      <td class="label">Title</td>
      <td><xsl:value-of select="fm:Title"/></td>
    </tr>
    <tr>
      <td class="label">Telephone Number</td>
      <td><xsl:value-of select="fm:Phone"/></td>
    </tr>
    <tr>
      <td class="label">Picture</td>
      <td><xsl:element name="img"><xsl:attribute
name="src"><xsl:value-of select="fm:Picture"/>
</xsl:attribute></xsl:element></td>
    </tr>
  </table>
</xsl:template>
<!--
  Don't display anything for the following elements.
-->
<xsl:template
match="fm:ERRORCODE|fm:DATABASE|fm:LAYOUT">
</xsl:template>
</xsl:stylesheet>
```

JavaScript scripting language example

Using HTML and a scripting language with your XML document can allow your web users to interact with the database after it has been downloaded. For example, a simple onClick scripting event handler can allow web users to click a button and see different records in the database.

This example demonstrates the use of the JavaScript scripting language with an XML document to publish the People.fp5 database on a web page. It starts with an HTML file, named People_form.htm, that 1) references the JavaScript library, FMP.js, 2) contains a simple HTML form with a table for the field and picture rows, and 3) builds and executes a CGI command to FileMaker Pro to find and download all the records in the People.fp5 database.



With JavaScript, you can manipulate the data in the XML document to display different records in the database—after the data has been downloaded

Note The FMP.js JavaScript library was created for the XML Inventory example, described next, which is a more complex and sophisticated example of the use of JavaScript and the W3C Document Object Model with your XML documents.

Once the people_form.htm page is loaded in the browser, the onLoad event handler performs the “initialize” function, creating an ActiveXObject and building the FMPFindRequest:

```
function initialize ( )
{
    var xmlDocument = new ActiveXObject ("Microsoft.XMLDOM");
    xmlDocument.async = false;
    var findRequest = new FMPFindRequest ("people.fp5", null, AND,
    false);
    if (xmlDocument.load(findRequest.getRelativeURL( ) ) )
    {
        foundSet = new FMPFoundSet (xmlDocument);
        populateFields( );
    }
    else
        alert ("Error retrieving records.");
}
```

The xmlDocument and findRequest variables are referenced from the FMP.js JavaScript library. The values in the new FMPFindRequest are people for the name of the database, null for no layout, AND for the find criteria's logical operator, and false for no XML DTD to be generated as a result of the request.

In the People_form.htm page, the nextRecord and previousRecord functions are assigned to the onClick event handler of the Next and Previous buttons respectively, so that when web users click the buttons, the function is performed and the fields are re-populated with new data.

```

function nextRecord( )
{
    if (foundSet.nextRecord( ))
        populateFields( );
}
function previousRecord( )
{
    if (foundSet.previousRecord( ))
        populateFields( );
}
function populateFields( )
{

    document.getElementsByName("Name") . item(0) . value =
        foundSet.getFieldByName ("Name");
    document.getElementsByName("Title") . item(0) . value =
        foundSet.getFieldByName ("Title");
    document.getElementsByName("Phone") . item(0) . value =
        foundSet.getFieldByName ("Phone");
    document.getElementsByName("Picture") . item(0) . src =
        foundSet.getFieldByName ("Picture");
}

```

Looking at the XML Inventory example

The XML Inventory example is a demonstration of a web-published database that uses the FMPXMLRESULT and FMPXMLLAYOUT grammars and Dynamic HTML (including JavaScript and the W3C Document Object Model) to display the XML data on the Web.

This example was designed to be viewed in the Internet Explorer 5.0 for Windows web browser. For information on new browsers that can be used to view the example, in FileMaker Pro, choose Help menu > FileMaker on the Web to go to the FileMaker Product support pages in your browser.

The example uses an inventory database for office equipment and provides two methods for displaying records from the database—in a list or in a detailed view of each record. Web site visitors can switch between List View and Detail View, and add, edit, delete, or find records in the database.

The XML Inventory example includes:

- a text file containing the JavaScript library used for this demonstration, named FMP.js
- the Inventory.fp5 database
- HTML files for viewing, searching, and adding records to the database



The Inventory.fp5 database used in the XML Inventory example

To view the XML Inventory example:

1. If necessary, copy the Inventory example files from the *FileMaker Developer CD* onto your hard disk.

Developer Extras\FileMaker, Inc>External FileMaker APIs/XML\Inventory Example

2. Place a copy of the Inventory Example folder and its files in the Web folder inside the FileMaker Developer application folder on your hard disk.

Web \ Inventory Example

Note For security, when actually publishing a database on the Web, you should not keep the database file in the Web folder unless you plan to administer it remotely. See “Opening password-protected databases remotely” on page 8-18 for information.

3. Open the Inventory.fp5 database in FileMaker Pro and make sure it's shared via the Web Companion.

See “Enabling the Web Companion” on page 8-3 and “Sharing the database via the Web” on page 8-5 for information.

4. In your web browser, type localhost (or your computer's IP address) followed by /inventory example/ and press Enter.

<http://localhost/inventory example/>

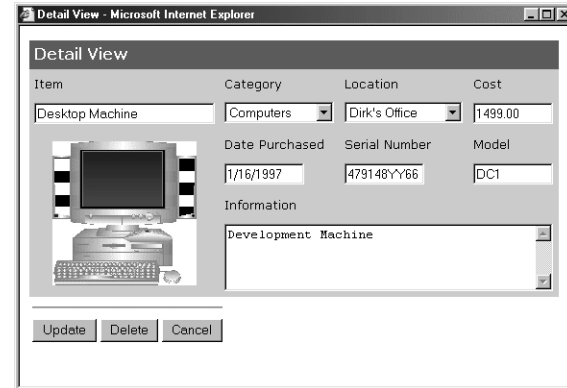
<http://17.17.17.17/inventory example/>

For information on setting up your computer as the localhost, see “Testing your site without a network connection” on page 8-17.

The XML Inventory example opens in List View in the browser.

5. Select a record and click the Detail View button.

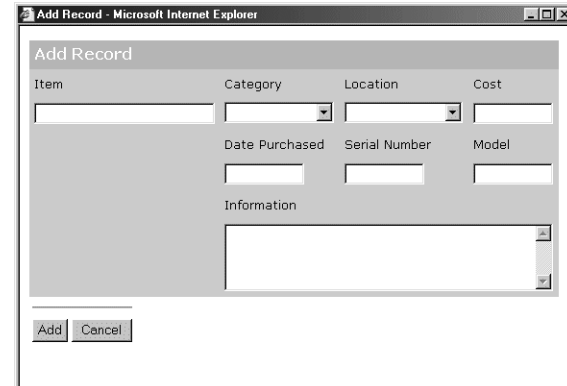
The currently selected record displays in a separate Detail View window. It is also possible to open the Detail View window by double-clicking a record item in the list. To close the Detail View window, click the Cancel button. To apply edits made to a record in Detail View, click the Update button.



Detail View of a selected record in the browser

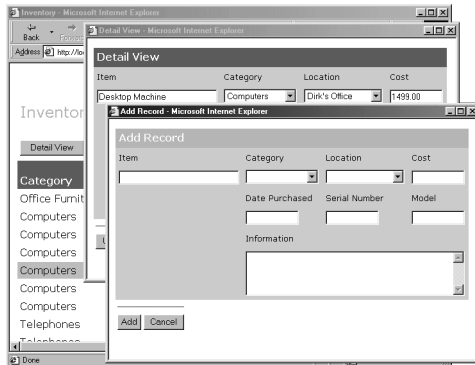
6. To add a record, click the Add button in the List View window.

A separate Add Record window opens.



7. Enter data for the new record and click Add to add it to the bottom of the list.

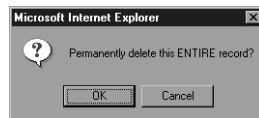
It is possible to have both the Detail View and Add Record windows open at the same time. The Detail View and Add Record windows will automatically close when a visitor clicks the Back or Forward button in the browser.



Detail View and Add Record windows open at the same time

8. To delete a record, click the Delete button in the Detail View or List View window.

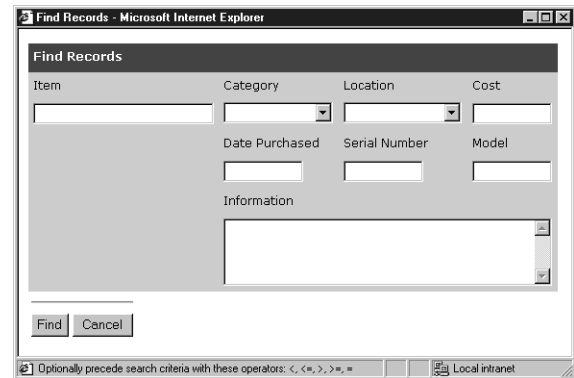
A confirmation dialog box provides options to cancel or delete the currently selected record.



9. Click Cancel.

10. To search for data, click the Find button in the List View window.

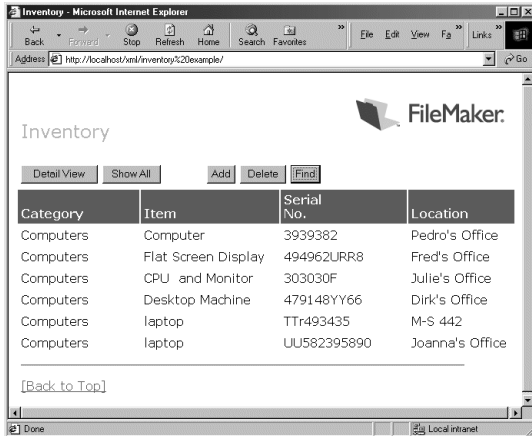
A detail view of an empty record opens in a separate window with a blinking insertion point in the Item field. When web users click inside the Item, Cost, Date Purchased, Serial Number, or Notes text boxes, a help string appears in the status bar at the bottom of the window. The default operator for a search is “begins with” or web users can type one of the <, <=, >, >=, or = operator symbols in these boxes.



Click in text boxes to display data entry instructions at bottom of window

11. Click in the Cost text box and type ≥ 2000 . Then click Find.

Any record containing the amount of \$2,000 or greater in the Cost field is found and displayed in List View.



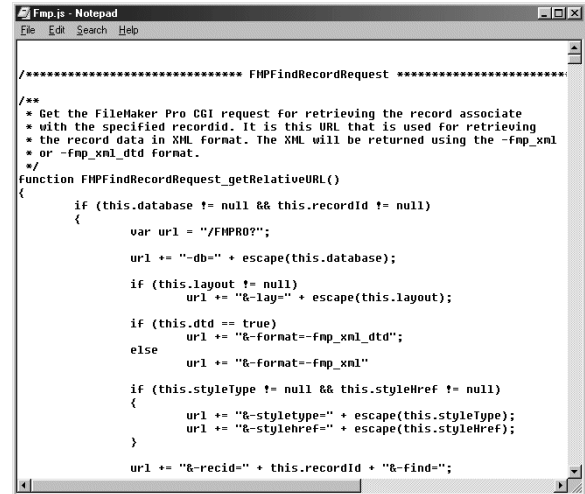
The found set of records is displayed in the List View window

12. In List View, click Show All to display all the records in the database, including any records you added.

If you want, you can examine the XML Inventory example's JavaScript library as you view the source for the example pages.

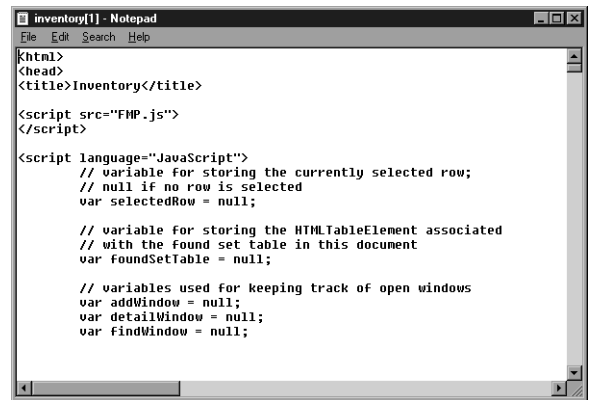
1. Start the Microsoft Notepad application (or a similar text editor), and choose File menu > Open. Locate and open the FMP.js file. (If necessary, choose All Files (*.*) as the Type.)

FileMaker Developer \ Web \ Inventory Example \ FMP.js



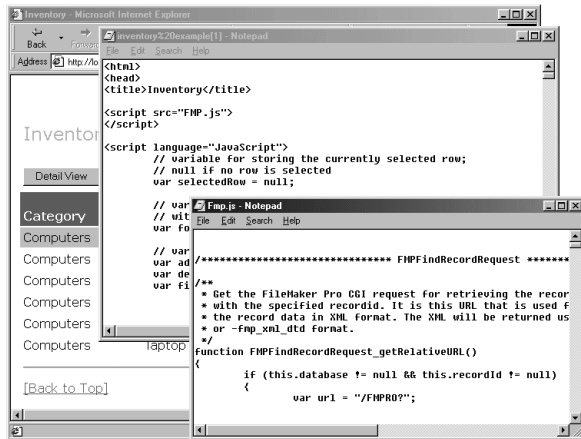
FMP.js text file containing the JavaScript library for this example

2. In the browser window, choose View menu > Source to see the source markup for the example pages in the Notepad application.



Source markup of the Default.htm page (List View)

The scripts and functions are well commented in the source and in the FMP.js library, providing information about each step. It's useful to arrange the windows so you can switch between the example pages in the browser, the source, and the script library.



Tip Use code snippets from the FMP.js JavaScript library for scripting your own web sites.

For more information about delivering your FileMaker Pro data using XML, in FileMaker Pro, choose Help menu > FileMaker on the Web.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 11

Using JDBC to deliver your data

If you're a Java programmer, you can use the FileMaker JDBC Driver with any Rapid Application Development (RAD) tool to visually create your FileMaker Pro database-aware Java application or applet.

The FileMaker JDBC Driver lets you directly access FileMaker Pro data using a RAD tool as you're building your code. Then, the Java application or applet that uses the FileMaker JDBC Driver can access FileMaker Pro data via the Web Companion.

About the JDBC examples

FileMaker Developer provides three examples of Java applications that use the FileMaker JDBC Driver to connect to a database. One example is a development-tool-independent Java application that was created using the basic Java classes and Sun Microsystems' Swing 1.1.1. The other two examples are Java front ends created with the development tools Corel's (Borland/Inprise) JBuilder 3.0 Professional for Windows and Symantec's Visual Cafe 4.0 Expert Edition for Windows.

For step-by-step instructions, see:

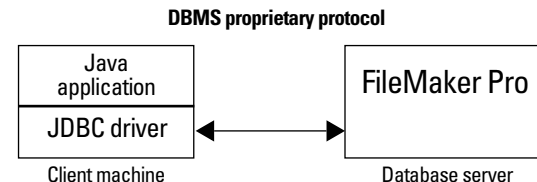
- "Example 1: Looking at the FileMaker Pro Explorer application" on page 11-8
- "Example 2: Creating the JBuilder Inventory application" on page 11-11
- "Example 3: Creating the Visual Cafe Inventory application" on page 11-15

For additional information and examples that use Java and JDBC for general data interchange or for publishing FileMaker Pro data on the Web, see the product support pages on the FileMaker, Inc. web site at www.filemaker.com. As a shortcut to the site, in FileMaker Pro, choose Help menu > FileMaker on the Web..

About JDBC

JDBC is a Java API for executing Structured Query Language (SQL) statements, the standard language for accessing relational databases. JDBC is a trademarked name and not an acronym—although it is thought of as standing for Java Database Connectivity because it is the ODBC (Open Database Connectivity) equivalent for Java. JDBC is a low-level interface, which means that it is used to call SQL commands directly. It is also designed to be used as a base for higher level interfaces and tools.

Your Java applet or application can talk directly to the database by using the JDBC driver to communicate with FileMaker Pro. Your SQL statements are delivered to the database and the results of those statements are sent back to you. The database can be located on another machine (the server machine) connected to the network, while your Java applet or application is located on your machine (the client machine). This is referred to as a client/server configuration.



Using the FileMaker JDBC Driver

You can use the FileMaker JDBC Driver with any Java compiler or RAD tool to connect with your database while you build the code for your Java application or applet. After the Java application or applet has been created, the FileMaker JDBC Driver must be present with the files or included within the code in order for the application or applet to communicate with the database.

To use the FileMaker JDBC Driver, your Java application or applet must register the driver with the JDBC driver manager and you must specify the correct JDBC URL from within the application or applet. You need the JDBC URL to make the connection to the database.

About the FileMaker JDBC Driver

The FileMaker JDBC Driver is a JDBC 1.2 API compatible driver designed to work with the Java Development Kit (JDK) 1.3. It is a Type 4 driver — a native protocol, pure Java driver that converts JDBC calls directly into the network protocol used by the database management system. This type of driver offers all the advantages of Java including automatic installation (for example, downloading the JDBC driver with an applet that uses it). The driver will work with JDK 1.3 and Java 2 as long as you only use JDBC 1.2 calls in a Java 2 environment.

Note Although the driver implements the entire JDBC 1.2 API, it cannot be classified as a true JDBC-compliant driver because it supports only a subset of SQL that matches the capabilities of FileMaker Pro, and is therefore not fully SQL-92 Entry Level compliant.

The FileMaker JDBC Driver is packaged as a Java archive file (with the .jar filename extension) containing a collection of class files. The archive file is named `Fmpjdbc12.jar`. The path to the file is:

```
FileMaker JDBC Driver > Fmpjdbc12.jar
```

The driver class and main entry point for the driver is named:

```
com.fmi.jdbc.JdbcDriver
```

Note The FileMaker JDBC driver as well as JDBC Examples were updated since the release of FileMaker Developer 5.0v1. The FileMaker JDBC driver now supports JRE 1.3. If you are using the example files from FileMaker Developer 5.0v1, please replace them with the example files on this CD. Three major changes to the JDBC driver are:

1. The FileMaker JDBC driver now supports JRE 1.3
2. Corrected a problem where blank repeating fields were not handled properly
3. Fixed problem where the specified column order was lost after the first query.

Using a JDBC URL to connect to your database

In Java, most resources are accessed through URLs (Uniform Resource Locators). A JDBC URL is used to identify the database so the FileMaker JDBC Driver can recognize and establish a connection with the database.

The JDBC URL consists of three main parts separated by colons:

```
jdbc:<subprotocol>:<subname>
```

The first part in the JDBC URL is always the JDBC protocol (“jdbc”). The *subprotocol* is the driver name or the name of the mechanism that supports multiple drivers. In this case, the subprotocol is `fmpro`, which is registered with Sun Microsystems, Inc. The *subname* is the IP address of the machine that is hosting FileMaker Pro.

The FileMaker JDBC Driver connects to FileMaker Pro through an HTTP connection. The subname in the JDBC URL includes an HTTP protocol (such as HTTP or HTTPS), an IP address or domain name, and an optional port number preceded by a colon.

```
jdbc:fmpro:http://1.184.21.234:80
```

The Web Server Connector allows you to encrypt and decrypt your data via HTTPS.

```
jdbc:fmpro:https://www.filemaker.com:80
```

Here is an example of registering the FileMaker JDBC Driver with the JDBC driver manager and connecting to FileMaker Pro via the Web Companion —where the open FileMaker Pro database is named Employees.fp5 and the JDBC URL is

```
jdbc:fmpro:http://localhost:
```

```
import java.sql.*;
class FMPJDBCTest
{
    public static void main(String[] args)
    {
        try
        {
            // register the FMPJDBC driver
            Class.forName("com.fmi.jdbc.JdbcDriver");

            // establish a connection to FileMaker Pro at a given IP
            // address
            Connection fmpConnection =
                DriverManager.getConnection("jdbc:fmpro:
                    http://localhost", "some_user", "some_password");

            Statement fmpStatement =
                fmpConnection.createStatement();

            // return a maximum of 50 records
            fmpStatement.setMaxRows(50);

            ResultSet resultSet = fmpStatement.execute("select \"last
                name\", \"first name\" from \"employees.fp5\" where
                department='engineering' order by \"last name\"");

            System.out.println("Engineering Department");
```

```
        while (resultSet.next())
            System.out.println(resultSet.getString("last name") +
                ", " + resultSet.getString("first name"));
        }
    }
    catch(ClassNotFoundException classNotFoundException)
    {
        System.out.println("Could not load driver");
    }
    catch(SQLException sqlException)
    {
        System.out.println("JDBC Error:" +
            sqlException.getMessage());
    }
}
}
```

Note This example is not meant to be compiled.

Specifying driver properties in the URL subname

You can specify the escape, fetchsize, user, and password driver properties in the subname of the JDBC URL. This is useful when you're using a RAD tool that doesn't support spaces, periods, or other non-letter characters.

```
jdbc:fmpro:http://17.184.21.234/properties?escape=%20&
fetchsize=100&user=fred&password=test
```

Note These are the same properties that could be passed to the connection when calling the `DriverManager.getConnection` method via the `Properties` parameter.

Property	Description
escape	A string containing the characters to be escaped in table name, field name, and layout name SQL identifiers. The driver will escape all identifiers returned via any method in the DatabaseMetaData class. This will allow RAD tools that don't support spaces and periods in SQL identifiers to work with any FileMaker Pro database. The driver will automatically escape all identifiers for you. See "Using a character escape" on page 11-7 for more information.
fetchsize	This property allows you to set the number of records that are retrieved by the driver at any one given time. This is important for result sets (such as a result set of 20000 records) that are too large to retrieve all at once without causing memory constraints and performance problems.
user	The user name for the connection
password	The password for the connection

SQL supported by the FileMaker JDBC Driver

The FileMaker JDBC Driver provides support for certain SQL statements, a RecordID pseudo column, a ModID pseudo column, DbOpen and DbClose pseudo procedures, character escaping, and FileMaker data type mapping to JDBC SQL and Java data types.

The following is a list of the SQL statements and definitions that are supported by the FileMaker JDBC Driver.

SQL statement	Definition
SELECT statement	<pre>SELECT { { * field_name ... } [, RECORDID [, MODID]] } FROM database_name [LAYOUT layout_name] [WHERE { predicate [{ { AND OR } predicate } ...] }] [ORDER BY { field_name [ASC DESC] } ...] Where <i>predicate</i> equals { field_name { = <> > >= < <= LIKE } { value ? } } {field_name IS NULL} {RECORDID = {value ?}}</pre>

SQL statement	Definition
INSERT statement	<pre>INSERT INTO database_name [LAYOUT layout_name] (field_name ...) VALUES ({ value NULL ? } ...)</pre>
UPDATE statement	<pre>UPDATE database_name [LAYOUT layout_name] SET { field_name = { value NULL ? } } ,... [WHERE { predicate [{ { AND OR } predicate } ...] }] Where <i>predicate</i> equals { field_name { = <> > >= < <= LIKE } { value ? } } {field_name IS NULL} {RECORDID = {value ?} [AND MODID = { value ? }] }</pre>
DELETE statement	<pre>DELETE FROM database_name [WHERE { predicate [{ { AND OR } predicate } ...] }] Where <i>predicate</i> equals { field_name { = <> > >= < <= LIKE } { value ? } } {field_name IS NULL} {RECORDID = {value ?} }</pre>
CALL stored procedure (a script) statement	<pre>{ CALL script_name (database_name [, { layout_name password }]) }</pre> <p>Where the outermost curly brackets { } are part of the CALL statement syntax.</p>

Note Items within square brackets [] are optional and a vertical bar | means "or." An ellipsis (...) indicates that the preceding part of the statement can be repeated any number of times. Periods and a comma (,...) indicate that the preceding part of the statement can be repeated any number of times with the individual occurrences separated by commas. The final occurrence should not be followed by a comma.

To update a specific repeating field or field in a portal, add a period and the number of the row to the end of the field name and enclose the field name in double quotation marks. For example, to update the third repetition of the Telephone field for a record in the Employees.fp5 database, specify the following:

```
UPDATE "Employees.fp5" SET "Telephone.3"='(555) 555-5555' WHERE
```

```
recordid=4
```

To add a specific repeating field or field in a portal, add a period and the number zero (0) to the end of the field name and enclose the field name in double quotation marks. For example, to add the City field to a portal in the Address relationship:

```
INSERT INTO "Employees.fp5" LAYOUT "Data Entry" ("Last Name",
"Address::City.0") VALUES ('Jones', 'San Jose')
```

Using DbOpen and DbClose pseudo procedures

The FileMaker JDBC Driver lets you open and close password-protected FileMaker Pro databases that have remote administration privileges and are located in the Web folder.

You only need to establish one connection to open or close your databases. Use “Admin” as the user name and the password that you specified in the Web Companion Configuration dialog box for remote administration. Once the databases are open, you’ll need one connection per unique database password to access the data.

Tip In order to minimize the number of connections, assign the same database password for all your databases.

Here is an example for opening and closing a password-protected database named “inventory.fp5,” where the remote administration user name is “Admin,” the remote administration password (set in the Web Companion Configuration dialog box) is “admin,” and the database password is “inventory.”

```
import java.sql.*;
/**
public class FMPJDBCSecurity
{
    public static void main(String[ ] args)
    {
        try
```

```
{
    // register the FMPJDBC driver
    Class.forName("com.fmi.jdbc.JdbcDriver");

    // establish a connection to FileMaker Pro for remote
    // administration purposes. The user name for remote
    // administration is always "Admin."
    Connection adminConnection =
        DriverManager.getConnection("jdbc:fmpro:
        http://localhost", "Admin", "admin");

    // create a statement for opening the "inventory.fp5"
    // database. Since the "inventory.fp5" database is
    // password protected, you also need to specify the
    // password in the call to the dlopen procedure.
    CallableStatement openDbStatement =
        adminConnection.prepareCall("{call
        dlopen(\"inventory.fp5\", \"inventory\")}");
    openDbStatement.execute();

    // establish a connection to FileMaker Pro for
    // retrieving data from the "inventory.fp5" database.
    // The "inventory.fp5" database uses FileMaker Pro
    // security, so you don't need to provide a user name.
    // A user name is only necessary for remote
    // administration or if the databases is protected via
    // the Web Security Database.
    Connection inventoryConnection =
        DriverManager.getConnection("jdbc:fmpro:
        http://localhost", null, "inventory");
    Statement inventoryStatement =
        inventoryConnection.createStatement();
    ResultSet inventoryData =
        inventoryStatement.executeQuery("select *
        from \"inventory.fp5\"");

    // create a statement for closing the "inventory.fp5"
    // database
    CallableStatement closeDbStatement =
        adminConnection.prepareCall("{call
        dbclos(\"inventory.fp5\")}");
    closeDbStatement.execute();
```

```

    }
    catch(ClassNotFoundException classNotFoundException)
    {
        System.out.println("Could not load driver");
    }
    catch(SQLException sqlException)
    {
        System.out.println("JDBC Error: " +
            sqlException.getMessage());
    }
}
}

```

Using the RecordID pseudo column

The FileMaker JDBC Driver provides a RecordID pseudo column (in place of a primary key used by other types of databases) that can be specified in the column name list of a SELECT statement or in the WHERE clause of SELECT, UPDATE or DELETE statements. This lets you guarantee that the statement will operate on a specific record.

All other columns are ignored when the RecordID pseudo column is used in a WHERE clause.

```
UPDATE "Employees.fp5" SET department='engineering' WHERE
recordid=4
```

Using the ModID pseudo column

Each record in a FileMaker Pro database has a corresponding modification ID (ModID) number that increases incrementally every time the record is modified. To detect modification collisions, the FileMaker JDBC Driver provides a ModID pseudo column that can be used in the WHERE clause of an UPDATE statement in conjunction with the RecordID. The Web Companion compares the ModID in the WHERE clause to the current ModID of the record and an error is returned if they do not match.

```

...
Connection connection = DriverManager.getConnection("jdbc:fmpro:
    http://localhost", "some_user", "some_password");
Statement statement = connection.createStatement();
// retrieve all of the records where department equals "engineering"
ResultSet resultSet = statement.executeQuery("SELECT recordid,
    modid, "last name", "first name" FROM \"Employees.fp5\" WHERE
    department='engineering'");
// create an UPDATE statement for changing the department to "software
// engineering"
PreparedStatement preparedStatement =
    connection.prepareStatement("UPDATE \"Employees.fp5\" SET
    department='software engineering' WHERE recordid=? AND
    modid=?");
while (resultSet.next())
{
    // set the recordid parameter
    preparedStatement.setString(1,
        resultSet.getString("RECORDID"));
    // set the modid parameter
    preparedStatement.setString(2, resultSet.getString("MODID"));
    // change the department from "engineering" to "software
    // engineering"
    preparedStatement.executeUpdate();
}
...

```


SQL statement examples

The following are some examples of SQL statements, some of which use RecordID and ModID pseudo columns, and DbOpen and DbClose pseudo procedures:

```
SELECT recordid, modid, "last name", "first name", department FROM
"Employees.fp5" WHERE "last name"='smith' AND "first name" = 'joe'
```

```
SELECT * FROM "Employees.fp5" WHERE recordid=4
```

```
SELECT recordid, modid, * FROM "employees.fp5"
```

```
SELECT "last name", "first name", "telephone::phone number" FROM
"employees.fp5" LAYOUT "personal info"
```

```
UPDATE "Employees.fp5" SET department='engineering' WHERE
recordid=4 AND modid=2
```

```
UPDATE "Employees.fp5" LAYOUT "personal info" SET
"telephone::phone number.2"='555-555-5555' WHERE recordid=4
```

```
DELETE FROM "Employees.fp5" WHERE recordid=4
```

```
{ CALL DbOpen("inventory.fp5", "some password") }
```

```
{ CALL DbClose("inventory.fp5") }
```

```
{ CALL FindManagers("employees.fp5") }
```

```
{ CALL SortByLastName("employees.fp5", "list view") }
```

Using a character escape

The FileMaker JDBC Driver supports escaping of lower ASCII characters in column and table name SQL identifiers. This is useful if your RAD tool doesn't support characters such as spaces in column names or periods in table names. The escape sequence starts with the dollar symbol (\$) and is followed by the two-digit hex value for the character (such as 2E for a period and 20 for a space).

```
employees.fp5 => employees$2Efp5
```

```
last name => last$20name
```

```
SELECT last$20name FROM employees$2Efp5
```

FileMaker data type mapping to JDBC SQL and Java data types

The FileMaker JDBC Driver uses the following mappings when converting FileMaker Pro data types to JDBC SQL types or to Java data types. (For information about these types, see the JDK 1.3 documentation web pages at www.javasoft.com.)

This FileMaker Pro data type Converts to this JDBC SQL type

This FileMaker Pro data type	Converts to this JDBC SQL type
TEXT	java.sql.Types.LONGVARCHAR
NUMBER	java.sql.Types.DOUBLE
DATE	java.sql.Types.DATE
TIME	java.sql.Types.TIME
CONTAINER	java.sql.Types.LONGVARBINARY

This FileMaker Pro data type Converts to this Java data type

This FileMaker Pro data type	Converts to this Java data type
TEXT	java.lang.String
NUMBER	java.lang.Double
DATE	java.sql.Date
TIME	java.sql.Time
CONTAINER	java.awt.Image
Repeating and related fields	com.fmi.jdbc.Array

FileMaker Pro support for Unicode characters

FileMaker Pro only supports the Windows Latin 1 and Macintosh character sets, which are a subset of Unicode. Therefore, any character data submitted to FileMaker Pro that contains characters not present in these character sets (such as certain math symbols) will not be stored properly in your database. FileMaker Pro inserts a question mark (?) for any character that it does not recognize.

About the FileMaker JDBC Driver interfaces and extensions

The FileMaker JDBC Driver implements all of the following JDBC interfaces:

- CallableStatement
- Connection
- DatabaseMetaData
- Driver
- PreparedStatement
- ResultSet
- ResultSetMetaData
- Statement

The following FileMaker Pro-specific extensions have been added:

This JDBC interface	Includes this FileMaker Pro extension
java.sql.DatabaseMetaData	com.fmi.jdbc.DatabaseMetaDataExt
java.sql.ResultSetMetaData	com.fmi.jdbc.ResultSetMetaDataExt

The following classes have been added in support of the FileMaker Pro extensions:

Class name	Description
com.fmi.fmpdb.FMPError	FileMaker Pro error codes
com.fmi.fmpdb.FMPLayoutField	Information associated with a field on a layout
com.fmi.fmpdb.FMPLayoutFieldEnumerator	Class for enumerating the fields on a layout
com.fmi.fmpdb.FMPLayoutMetaData	Metadata for a given layout
com.fmi.jdbc.Array	Class used to represent repeating and related fields

The API documentation for these standard interfaces and the FileMaker extensions is included in HTML format on the *FileMaker Developer CD*:

Developer Extras>FileMaker, Inc>External FileMaker APIs>FileMaker JDBC Driver>JDBC Documentation

Note Retrieving large text fields with the `ResultSet.getBytes` method and sending large amounts of text back to FileMaker Pro using an `INSERT` or `UPDATE` statement with MRJ 2.2 can cause sluggish performance.

Example 1: Looking at the FileMaker Pro Explorer application

This developer-tool-independent example is a Java application used for displaying FileMaker Pro database information, similar to the Windows Explorer and Mac OS Finder applications. You can use the FileMaker Pro Explorer application along with the FileMaker JDBC Driver to view any open database on any computer that's shared via the Web Companion, by specifying the JDBC URL that includes the IP address of the computer where FileMaker Pro is running. You can view the application's source code in any text editor or Java editing tool.

The application was created using the basic Java classes to display a database tree, and FileMaker Pro-specific extensions have been added to provide detailed information about the fields and layouts. The user interface was created using the Swing 1.1.1 class library—an add-on to the Java Development Kit (JDK) 1.3.

For information on the Swing class library, go to the Sun Microsystems web site at www.javasoft.com.

Setup requirements

Included with the example is the swingall.jar file, on the *FileMaker Developer CD*.

Developer Extras>FileMaker, Inc>External FileMaker APIs>FileMaker JDBC Driver>JDBC Examples>FileMaker Explorer

To view the example on Windows machines, you need:

- Java.exe (included with JDK 1.3 for Windows) or equivalent Java virtual machine installed in the system path on your computer

To view the example In Mac OS machines, you need:

- MRJ 2.2 or equivalent Java virtual machine installed on your computer

Install the example and the FileMaker JDBC Driver

If necessary, install the FileMaker Explorer example and the FileMaker JDBC Driver.

Developer Extras\FileMaker, Inc\External FileMaker APIs\FileMaker JDBC Driver \JDBC Examples\ FileMaker Explorer

Developer Extras\FileMaker, Inc\External FileMaker APIs\ FileMaker JDBC Driver\Fmpjdbc12.jar

Open and share your databases via the Web

1. In FileMaker Pro, open any FileMaker Pro database, such as the Inventory.fp5 database located in the JBuilder example folder:

Developer Extras\FileMaker, Inc\External FileMaker APIs\ FileMaker JDBC Driver\JDBC Examples \Builder 3.0 Professional \Inventory.fp5

2. Choose File menu > Sharing, verify that Web Companion is selected, and click OK to share the database on the Web.

For information about setting up the Web Companion so that it's already selected in this dialog box, see "Enabling the Web Companion" on page 8-3.

Run the FileMaker Pro Explorer application

The FileMaker Pro Explorer application is located in the FileMaker Explorer folder:

Developer Extras\FileMaker, Inc\External FileMaker APIs\FileMaker JDBC Driver \Examples\FileMaker Explorer\FileMakerExplorer.bat

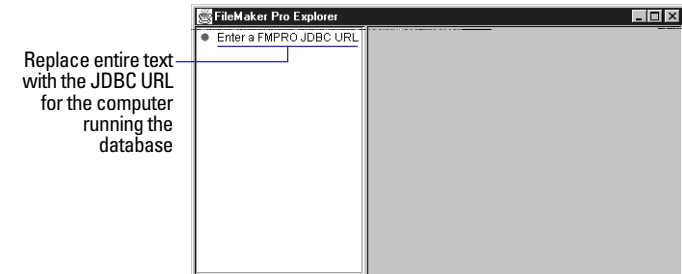
Developer Extras\FileMaker, Inc\External FileMaker APIs\FileMaker JDBC Driver\Examples\FileMaker Explorer\FileMakerExplorer

1. Start the FileMaker Pro Explorer application by doing one of the following:

- Windows: Choose Start menu > Run and locate and select the FileMakerExplorer.bat file. Then, add the location of the JDK for running the application at the end of the command line:

"full path\FileMakerExplorer.bat" c:\jdk1.3

- Mac OS X: Double-click the application icon to start FileMaker Pro Explorer.

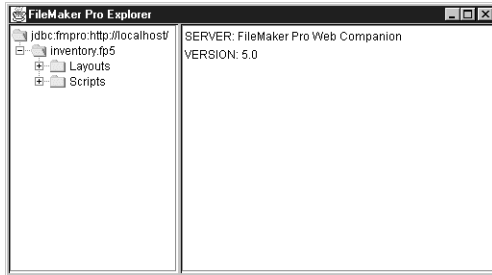


2. Click the root node of the tree in the left side of the FileMaker Pro Explorer window to select it, and replace the entire text with the JDBC URL for the computer that is running the database, for example, type `jdbc:fmp:pro:http://localhost/`. Then press Enter.

See “Using a JDBC URL to connect to your database” on page 11-2 for information.

3. Select Inventory.fp5 (or other database name) on the left side of the window to see information about it on the right side.

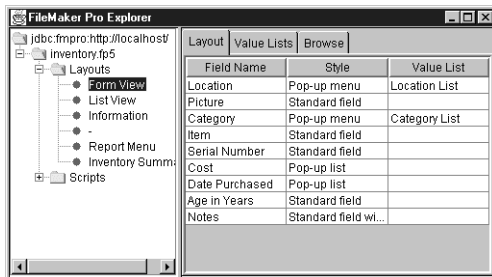
4. Expand the Inventory.fp5 node (folder) to expose the Layouts and Scripts nodes.



5. Select Layouts to display the names of the layouts for the Inventory.fp5 database on the right side of the window.

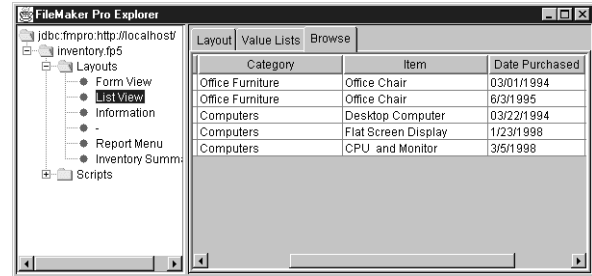
6. Expand the Layouts node to display a leaf node for each layout.

7. Select the Form View node to display the fields for that layout on the right side of the window.



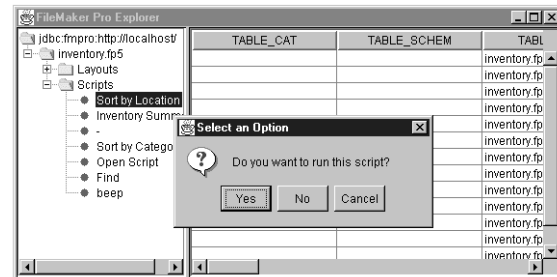
8. Click the Browse tab to display the first five records in the Inventory.fp5 database.

The columns correspond to the fields available for the selected layout.



9. Select the Scripts node to display the names of scripts for the Inventory.fp5 database on the right side of the window.

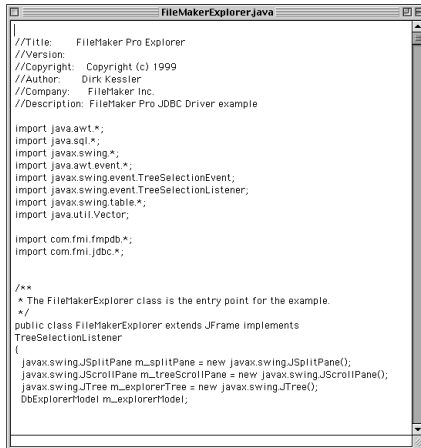
When you select a script name, a warning dialog box appears. This gives you the opportunity to not run the script, which is especially important if a database includes a script for deleting or modifying records.



View the source code of the example

1. Start your text editor (such as Notepad or SimpleText) or Java development tool.

2. Open the main class file, FileMakerExplorer.java, in the FileMaker Explorer folder.



```

FileMakerExplorer.java
//Title: FileMaker Pro Explorer
//Version:
//Copyright: Copyright (c) 1999
//Author: Dirk Kessler
//Company: FileMaker Inc.
//Description: FileMaker Pro JDBC Driver example

import java.awt.*;
import java.sql.*;
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.event.TreeSelectionEvent;
import javax.swing.event.TreeSelectionListener;
import javax.swing.table.*;
import java.util.*;

import com.fmi.fmpdb.*;
import com.fmi.jdbc.*;

/**
 * The FileMakerExplorer class is the entry point for the example.
 */
public class FileMakerExplorer extends JFrame implements
TreeSelectionListener
{
    javax.swing.JSplitPane m_splitPane = new javax.swing.JSplitPane();
    javax.swing.JScrollPane m_treeScrollPane = new javax.swing.JScrollPane();
    javax.swing.JTree m_explorerTree = new javax.swing.JTree();
    DbExplorerModel m_explorerModel;
}

```

The source code is well commented, describing the methods for each class used in this example.

Example 2: Creating the JBuilder Inventory application

This example demonstrates how to build a Java front end to a FileMaker database using the development tool, JBuilder 3.0 Professional for Windows and the FileMaker JDBC Driver. This example uses a modified version of the Asset Management.fp5 database that ships with FileMaker Pro. The following steps are for creating a Java application that accesses the database, renamed Inventory.fp5.

Install the example and FileMaker JDBC Driver

If necessary, install the JBuilder folder of example files and the FileMaker JDBC Driver.

Developer Extras\FileMaker, Inc\External FileMaker APIs\FileMaker JDBC Driver\JDBC Examples\JBuilder 3.0 Professional

Developer Extras\FileMaker, Inc\External FileMaker APIs\FileMaker JDBC Driver\Fmpjdbc12.jar

The JBuilder 3.0 Professional example folder contains the database file used in this example and all of the completed files generated by the JBuilder wizard for the application.

Set up JBuilder to use the FileMaker JDBC Driver

1. In a text editor (such as Notepad), open the Jbuilder.ini file from the bin folder inside the JBuilder 3.0 Professional application folder. (Please make a backup copy of this file before proceeding with these instructions.)

2. In the [Java_VM_Properties] section, add the path of the Fmpjdbc12.jar file to the end of the Djava.class.path line.

For instructions on installing the FileMaker JDBC driver into the JBuilder environment, see “Installing and setting up JBuilder for database applications” in the online JBuilder Help or go to www.borland.com/devsupport/jbuilder/.

Open and share the Inventory.fp5 database

1. In FileMaker Pro, open the Inventory.fp5 file in the JBuilder 3.0 Professional folder:

Developer Extras\FileMaker, Inc\External FileMaker APIs\FileMaker JDBC Driver\JDBC Examples\JBuilder 3.0 Professional\Inventory.fp5

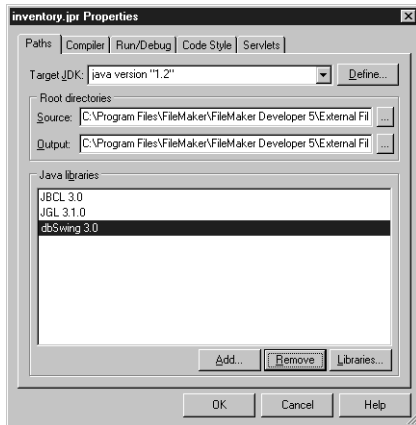
2. Choose File menu > Sharing, verify that Web Companion is selected, and click OK to share the database on the Web.

For information about setting up the Web Companion so that it’s already selected in this dialog box, see “Enabling the Web Companion” on page 8-3.

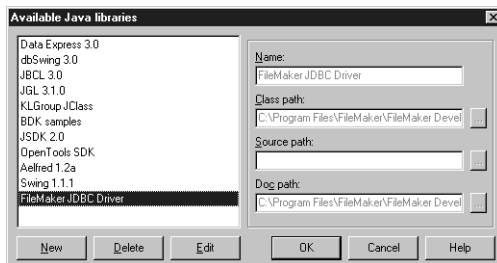
Start a new JBuilder project

1. Start JBuilder 3.0 Professional for Windows.
2. In JBuilder, choose File menu > New Project.

- In the dialog box, specify the name and location for the project file (JBuilder Inventory).
- Choose Project menu > Properties.



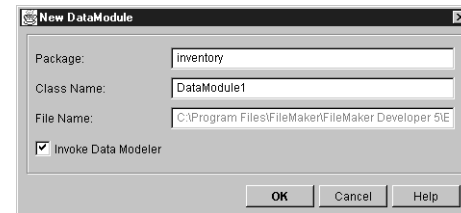
- Click Libraries in the Properties dialog box.
- In the Available Java Libraries dialog box, select the Fmpjdbc12.jar file. (If the FileMaker JDBC Driver does not appear in this dialog box, click New to locate the file and add it to the list.) Then click OK.



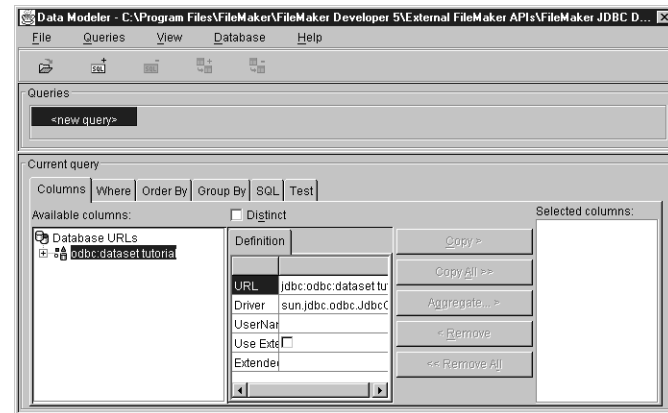
- Click OK to close the Properties dialog box.

Create the data module

- In JBuilder, choose File menu > New.
- In the New dialog box, select the Data Module icon and click OK.
- In the New Data Module dialog box, make sure that inventory is in the Package text box and that the checkbox for Invoke Data Modeler is selected.



- Click OK to open the Data Modeler.



Design the data module

- In the Data Modeler, choose Database menu > Add Connection URL.



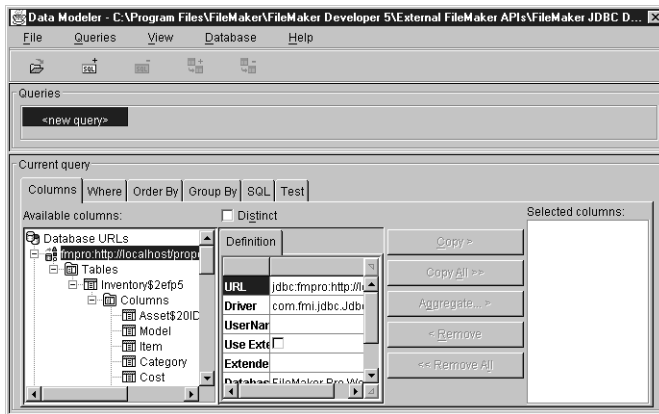
2. In the Driver text box, type `com.fmi.jdbc.JdbcDriver`.

3. In the URL text box, type `jdbc:fmpro:http://localhost/properties?escape=%20`.

Because JBuilder's Application Generator doesn't allow for periods in table names or spaces in column names for certain operations, you need to add escape properties for them in the subname of the URL.

4. Click OK to close the Add Connection URL dialog box.

5. In the Data Modeler, click the + symbol to expand the URL node that you added.



6. In the User Authentication dialog box, leave the User Name and Password text boxes blank (the database is not password protected) and click OK.

JBuilder connects to FileMaker Pro.

7. In the Data Modeler, click the + symbol to expand Tables.

8. Click the + symbol to expand the Inventory\$2efp5 table (now encoded with the period escaped).

9. Click the + symbol to expand Columns.

10. Select each of the following columns and copy it to the right side of the Data Modeler.

- Item
- Category
- Location
- Cost
- Date Purchased
- Picture
- Serial Number
- Information

Test the data module

1. In the Data Modeler, click the Test tab and click Execute Query.

The data and all of the images from the Inventory.fp5 database are downloaded into the data module.

2. Choose File menu > Save.

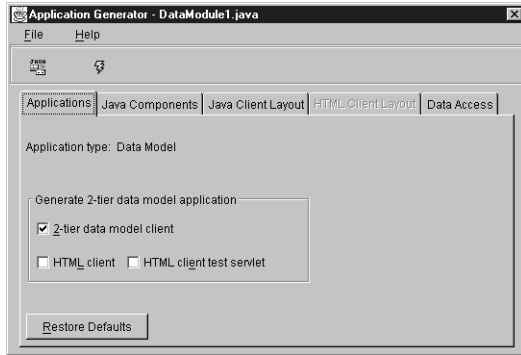
JBuilder saves the data module and automatically adds it to your project.

3. Choose File menu > Exit.

4. Click Yes to invoke the Application Generator.

Generate the application

1. In the Application Generator, click Restore Defaults and make sure that 2-tier data model client is selected.



2. Click the Java Client Layout tab.

Because JBuilder requires a unique row identifier for updating and deleting rows, it automatically adds the RecordID pseudo column to the list of output columns, which you probably don't want displayed in your application.

3. Deselect the Layout check mark for the RecordID identifier.

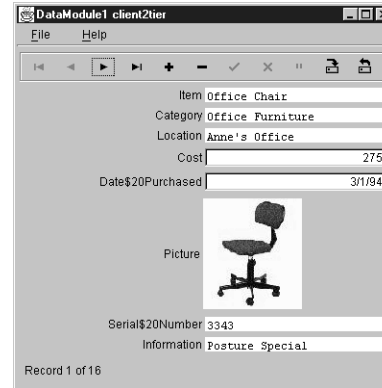
4. Choose File menu > Generate.

5. Click Generate.

The Application Generator generates the source code and adds the generated files to your project.

6. Click Close All.

7. Choose Run menu > Run Application to execute the example application.



The field attributes used for the example from the Inventory.fp5 database are described in this table:

Field type	Field name	Field attribute
Text	Item	User defined entry field
Text	Category	Value List/Category List (Pop-up menu) Office Furniture Computers Telephones Vehicles Copier Printer AV
Text	Location	Value List/Location List (Pop-up menu) Fred's Office Dirk's Office Pedro's Office Anne's Office Julie's Office Ruth's Office Joanna's Office Business Center
Number	Cost	User defined entry field

Field type	Field name	Field attribute
Date	Date Purchased	User defined entry field
Container	Picture	Graphic import
Text	Serial Number	User defined entry field
Text	Information	User defined entry field

Example 3: Creating the Visual Cafe Inventory application

This example demonstrates how to build a Java front end to a FileMaker database using the development tool Symantec's Visual Cafe Expert Edition for Windows and the FileMaker JDBC Driver. The following steps are for creating a Java application that accesses the inventory_db database.

Note Because the Visual Cafe DataBound Project Wizard does not support spaces in database filenames or field names, or periods and spaces in table names, this example uses a modified version of the Asset Management.fp5 database that ships with FileMaker Pro, called inventory_db (with no filename extension).

Install the example and the FileMaker JDBC Driver

If necessary, install the Visual Cafe example and the FileMaker JDBC Driver.

Developer Extras\FileMaker, Inc\External FileMaker APIs\FileMaker JDBC Driver \JDBC Examples \Visual Cafe 4.0 Expert Edition

Developer Extras\FileMaker, Inc\External FileMaker APIs\FileMaker JDBC Driver\Fmpjdbc12.jar

The Visual Cafe folder contains the modified database file used in this example and all of the completed files generated by the DataBound Project Wizard for the application.

Set up Visual Cafe to use the FileMaker JDBC Driver

1. Start Visual Cafe 4.0 Expert Edition for Windows.
2. Choose Tools menu > Environment Options and click the Internal VM tab. Then click New for the Classpath Settings.
3. Locate the Fmpjdbc12.jar file and add it to the SC.INI CLASSPATH list to add the FileMaker JDBC Driver to the Visual Cafe environment.

Open and share the inventory_db database

1. In FileMaker Pro, open the inventory_db file in the Visual Cafe example folder:

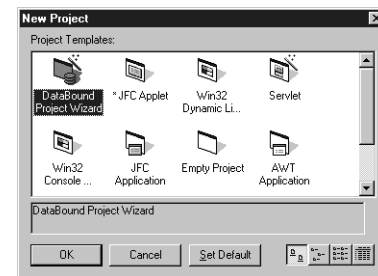
Developer Extras\FileMaker, Inc\External FileMaker APIs\FileMaker JDBC Driver \JDBC Examples \Visual Cafe 4.0 Expert Edition \inventory_db

2. Choose File menu > Sharing, verify that Web Companion is selected, and click OK to share the database on the Web.

For information about setting up the Web Companion so that it's already selected in this dialog box, see "Enabling the Web Companion" on page 8-3.

Create a new Visual Cafe project

1. In Visual Cafe, choose File menu > New Project.
2. Select the DataBound Project Wizard template and click OK.



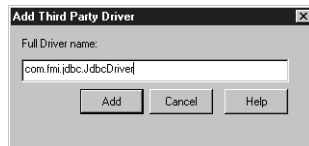
3. In the DataBound Project Wizard, select **Application** for the project type and click **Next**.

4. Select **Define another data source** and click **Next**.

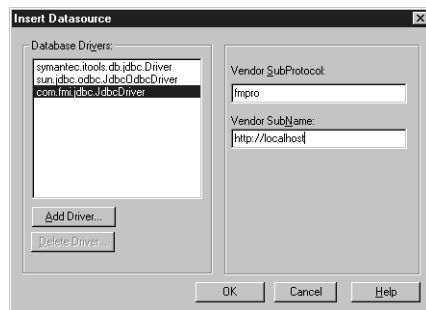
5. In the DataBound Project Wizard, click **New**.

6. In the Insert Datasource dialog box, click **Add Driver**.

7. In the Add Third Party Driver dialog box, type `com.fmi.jdbc.JdbcDriver` and click **Add**.



8. In the Insert Datasource dialog box, select `com.fmi.jdbc.JdbcDriver`, type `fmpro` in the Vendor SubProtocol box, type `http://localhost` in the Vendor SubName box (or the IP address of the computer hosting the `inventory_db` database), and then click **Add Driver**.



9. In the DataBound Project Wizard, select `com.fmi.jdbc.JdbcDriver` in the list of data sources and click **Next**.

10. In the User Authentication dialog box, click **OK**.

The `inventory_db` database is not password protected.

11. In the DataBound Project Wizard, select `inventory_db` in the list of database tables and views, and click **Next**.

12. Click **Clear** to move all the used columns to the Available Columns list.

13. Select the following columns and move them to the Used Columns list by selecting each one and clicking **Move**.

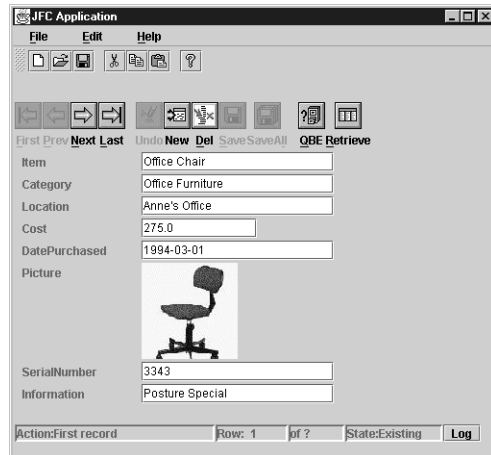
- Item
- Category
- Location
- Cost
- Date Purchased
- Picture
- Serial Number
- Information

14. Click **Next**.

15. In the DataBound Project Wizard, select **Picture** and choose **ImageViewer** from the Component list to change the UI component used for displaying the **Picture** column.

16. Click **Finish** to generate the code.

17. Choose **Project menu > Execute** to run the application.



The field attributes used for the example from the inventory_db database are described in the following table:

Field type	Field name	Field attribute
Text	Item	User defined entry field
Text	Category	Value List/Category List (Pop-up menu) Office Furniture Computers Telephones Vehicles Copier Printer AV

Field type	Field name	Field attribute
Text	Location	Value List/Location List (Pop-up menu) Fred's Office Dirk's Office Pedro's Office Anne's Office Julie's Office Ruth's Office Joanna's Office Business Center
Number	Cost	User defined entry field
Date	Date Purchased	User defined entry field
Container	Picture	Graphic import
Text	Serial Number	User defined entry field
Text	Information	User defined entry field

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 12

Understanding external function plug-ins

If you are a C or C++ programmer and familiar with advanced calculations in FileMaker Pro, you can create external function plug-ins that extend the feature set of FileMaker Pro—including calculation formulas that take advantage of recursion and looping, or that hook into other programming interfaces. Users can enable your plug-ins in FileMaker Pro and use your external functions in their calculation fields and scripts.

FileMaker Pro plug-ins must be registered with FileMaker, Inc. The FileMaker, Inc. web site (www.filemaker.com) includes a plug-in registration form and a database of all the registered plug-ins. You can browse this database to get an idea of what kind of plug-ins already exist and use it to list your own plug-ins. See “Registering your plug-ins” on page 12-13 for more information.

About external functions

The *FileMaker Developer* CD includes an example plug-in project that you can modify to include your own external functions. Users can access your plug-ins through the FileMaker Pro Specify Calculation dialog box.

Follow these general steps to prepare your custom plug-ins:

1. Edit the example plug-in files to add your custom programming code.
2. Compile and test the customized plug-in.
3. Register your functions with FileMaker, Inc.
4. Install the compiled plug-in file for your users.

To access your custom functions, your users do the following:

1. Enable your plug-in through the Application Preferences dialog in FileMaker Pro.
2. Configure your plug-in, if required.
3. Define or edit a calculation field in FileMaker Pro.
4. In the Specify Calculation dialog box, choose External(function_name) as the calculation formula.

About the plug-in example file

The example plug-in project is designed to illustrate what a complete FileMaker Pro plug-in looks like. You can compile the example project files to create a plug-in with several external functions that users can access through the Specify Calculation dialog box in FileMaker Pro. You can examine and modify the source code of these examples and templates in any text editor.

The plug-in example includes five useful external functions (see “Description of the FMExample plug-in’s external functions” on page 12-4).

The plug-in example files include all the source code required to compile the plug-in for the Windows, Mac OS, and Mac OS X platforms. In addition to the plug-in source code, FileMaker Developer includes project files for CodeWarrior 4, CodeWarrior 6, and Microsoft Developer Studio.

The example plug-in files in Mac OS are located in the Developer Extras\FileMaker, Inc\External FileMaker APIs\External Function Plug-in\Output\FAT folder on the *FileMaker Developer* CD.

The example plug-in files in Windows are located in the Developer Extras\FileMaker, Inc\External FileMaker APIs\External Function Plug-in\Output\x86 folder on the *FileMaker Developer CD*. The plug-in example source code files are located in subfolders in the FMExample plug-in folder. The following tables describe some of the files.

Contents of the FMExample Plugin folder

File/Folder	Description
FMExampleCW4.mcp	CodeWarrior 4 project file
FMExampleCW6.mcp	CodeWarrior 6 project file
FMExampleVC.dsp	Microsoft Developer Studio project file
FMExampleVC.dsw	Microsoft Developer Studio workspace file
(Output) folder•	The (Output) folder contains one subfolder for each of the possible target platforms. The compiled plug-in is automatically placed in the appropriate subfolder.

Contents of the EFP API folder

The EFP API folder contains the header and source files for FileMaker application program interface (API) code. This code is proprietary and non-redistributable. See “Abiding by the license agreement” on page 1-5 for details.

File/Folder	Description
FMFlags.h	Contains statements to set various compile-time flags.
FMExtern.h	Contains the FileMaker API function definitions.
FMExtern.c	Contains the FileMaker API functions source code.

Contents of the Resources folder

The Resources folder contains the resource files for Windows and Mac platforms.

File/Folder	Description
FMResource.h	Contains the plug-in resources definitions.
FMExample.r•	Contains the resources source code for Mac platforms.
FMExample.rc•	Contains the resources source code for Windows platforms.

Contents of the Source folder

The Source folder contains the header and source files that make up the example plug-in code.

File/Folder	Description
TargetPrefix.Classic•	Contains flag definitions for the Mac Classic platform.
TargetPrefix.Carbon	Contains flag definitions for the Mac OS X platform.
TargetPrefix.x86	Contains flag definitions for the Windows platform.
FMUtils.h	Contains definitions for plug-in support functions.
FMUtils.c	Contains source code for plug-in support functions.
FMPrefs.h•	Contains definitions for plug-in configuration functions.
FMPrefs.c•	Contains source code for plug-in configuration functions.
FMInitIdle.h•	Contains definitions for plug-in initialization, idle, and shut down functions.
FMInitIdle.c•	Contains source code for plug-in initialization, idle, and shut down functions.
FMMain.h	Contains definitions for plug-in entry point functions.

File/Folder	Description
FMMain.c•	Contains source code for plug-in entry point functions.
FMFunc.h	Contains definitions for external functions.
FMFunc.c	Contains source code for external functions.

Installing, enabling, and configuring the example plug-in

External function plug-in files must be installed in the appropriate folder and enabled in FileMaker Pro before they can be used. Some plug-ins must also be configured by the user.

To install a plug-in:

- On Windows machines, drag the plug-in file into the System folder inside the FileMaker Developer 6 folder. In Windows, the plug-in extension *must* be .FMX.
- On Mac OS machines, drag the plug-in file into the FileMaker Extensions folder inside the FileMaker Developer 6 folder.

To enable a plug-in:

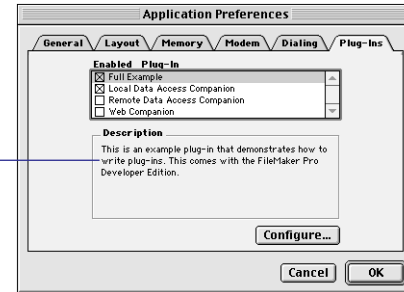
1. Choose Edit menu > Preferences > Application.

Mac OS X: choose FileMaker Developer application menu > Preferences > Application.

2. In the Application Preferences dialog box, click the Plug-ins tab.
3. Select the plug-in in the list.

A plug-in will not appear in the list if it's not installed in the correct folder inside the FileMaker Developer application folder.

Plug-in description text is defined in a resource string. See "Required resource files" on page 12-7.



Select a plug-in to enable it

To configure a plug-in:

1. Select the plug-in in the Application Preferences dialog box.
2. Click Configure.

The Configure button is only available when the plug-in is selected and if the sixth character in the feature string is "Y". See "Feature string syntax" on page 12-8.

3. Click OK to close the configuration dialog box.
4. Click Done to close the Application Preferences dialog box.

Description of the FMExample plug-in's external functions

The FMExample plug-in provided in the Microsoft Visual C++ and Code Warrior Pro example projects adds the following external functions to FileMaker Pro.

Function's name and parameter	Description of external function
Xpl-Version, ""	<p>This function returns the version of the plug-in. It has no parameters.</p> <p>Note A version function similar to the one provided in the example plug-in is required for every FileMaker Pro plug-in.</p>
Xpl-NumToChar, integer	<p>This function returns the ASCII character that corresponds to the integer (1-255) passed to it.</p>
Xpl-CharToNum, character	<p>This function returns the ASCII integer value of the character passed to it.</p>
Xpl-Format, <a string of numbers>	<p>This function returns the parameter as a text string formatted as specified in the configuration dialog box. Use this function to format numbers such as telephone numbers, postal codes, and so on.</p> <p>Formatting proceeds from right to left. Each # symbol in the format string is replaced by the next number in the parameter string. All remaining # symbols are replaced with zeros. The Xpl-Format function is not designed to handle strings containing alphabetical characters. For example, if the parameter is a phone number like 1-800-ABC-DEFG, the function ignores the alpha characters and returns something like (000) 000-1800.</p> <p>If the parameter contains more than 249 characters or if there are more characters than there are # symbols in the format string, then the function returns a -01.</p>

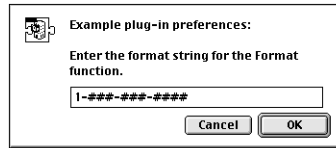
Function's name and parameter	Description of external function
Xpl-NumWords, <numbers in a floating format, like "44.345" up to 9,999,999.99>	<p>This function returns a number in bank check format. - For example 44.345 returns Forty-Four Dollars and - 34 Cents. -</p> <p>All digits beyond the third decimal place and any alphabetical characters are ignored.</p> <p>Note The source code for the Xpl-NumWords function is based on the calculation formula described in the TechInfo article "Converting Numbers to Words or Numbers to Text." By writing the formula into the code, this FMExample plug-in demonstrates how an external function can save users hours of database development time.</p> <p>For a description of the formula, go to www.filemaker.com/support. Click on the TechInfo database link. In the search dialog box on the Web site, choose Product: FileMaker. In the Article Number box, type 104580.</p>

Using the example plug-in

To access the external functions, do the following:

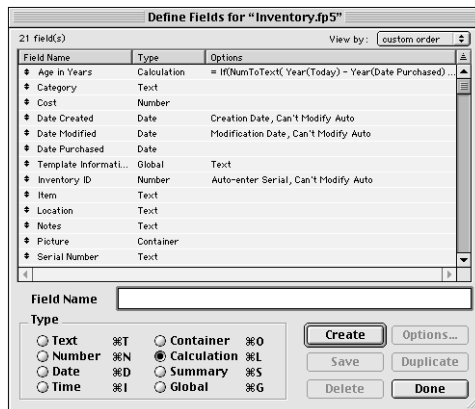
1. Install the FMExample.fmx file (Windows) or FMExample file (Mac OS). See "Installing, enabling, and configuring the example plug-in" on page 12-3.
2. In the Application Preferences dialog box, select FMExample to enable the plug-in. Because the example plug-in includes a function that requires configuration, the Configure button is enabled.
3. Click Configure.

The configuration dialog box that appears depends on how the plug-in source code was written. The XPL-Format function in the example plug-in displays the following configuration dialog box.

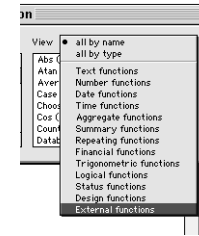


Dialog box that appears when you configure the example plug-in

4. Click **OK** to use the default format, or type a new format. The “#” symbols are replaced by numbers. All other text in the format string is retained as is.
5. Click **Done** to close the Application Preferences dialog box.
6. In FileMaker Pro, choose **File menu > Define Fields**.

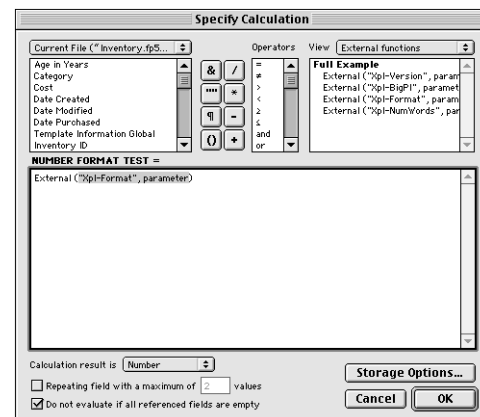


7. Type a name for a new calculation field in the **Field Name** box.
8. For **Type**, select **Calculation**, and click **Create**.
9. In the **Specify Calculation** dialog box, choose **External Functions** from the **View** pop-up menu.



10. Double-click an external function in the list that begins with the function prefix "Xpl" to add it to the formula box.

Note The name of the External Function FMExample will show in bold and the specific functions will be underneath that.



All external function calls require the name of the external function to call and the function's parameter value, even if the value is null ("").

11. Replace the word “parameter” with the required parameter for the function. If no parameter is required, type two double-quotes (“”).

12. Continue to build the formula as desired and click **OK** when you're done.

13. Click **Done** to close the Define Fields dialog box.

Customizing the plug-in example

The plug-in example in FileMaker Developer is designed to be easily modified so you can add your own custom functions. You need to modify the following:

- version and function name information in `FMExample.r`, and `FMExample.rc`
- configuration function in `FMPrefs.c`
- registry information in `FMMain.h`
- external function definitions in `FMFunc.h`
- external function coding in `FMFunc.c`

The following tables describe the modifications you must make to the plug-in example files to create a custom external function plug-in.

Customizing `FMExample.r`

<code>resource kFMEX_RES_CREATOR</code>	Specify the creator string value.
<code>resource 'vers'</code>	Specify the version values.
<code>resource 'STR#'</code> <code>kFMEX_RES_STRINFOID</code>	Specify the plug-in name and description. Modify the feature string as appropriate.
<code>resource 'STR#'</code> <code>kFMEX_RES_FUNCNLISTID</code>	Include the names of all external functions in the plug-in.

Customizing `FMExample.rc`

<code>//Version</code>	Modify the version variables as appropriate.
<code>//Dialog</code>	Specify a custom caption for the dialog.
<code>//Icon</code>	Change icon location variable to point to your icon.

<code>//String Table•</code>	Edit the String Table strings as appropriate, including the plug-in name, the Feature String, and the list of function names in the plug-in. See “Required resource files” on page 12-7, and “Feature string syntax” on page 12-8.
------------------------------	--

Customizing `FMPrefs.c`

<code>Do_PluginPrefs•</code>	Add your configuration code to the <code>Do_PluginPrefs</code> function.
------------------------------	--

Customizing `FMMain.h`

<code>#define k_Pref_Registry_Path_Value•</code>	Place your preferences in the registry under your own software company or product name.
--	---

Customizing `FMFunc.h`

<code>kVersionLength</code>	Set value to equal the length of <code>kVersionString</code> .
<code>kVersionString</code>	Specify the plug-in name and version.
<code>Function declarations•</code>	Include your function declaration statements as appropriate.

Customizing `FMFunc.c`

<code>Do_ExternalFunction•</code>	Modify the case statements to call your functions, as appropriate. The first function in the function list of the resource file is numbered 0.
<code>Function Statements•</code>	Add your function code following the <code>Do_ExternalFunction</code> function. The <code>PluginVersion</code> function is <i>required</i> for all plug-ins. Do not delete this function.

Requirements for writing an external function plug-in

FileMaker Pro plug-ins are most useful when they contain a single function or a set of functions with similar features. When you design your plug-in, keep in mind that most database developers who use your plug-in may not understand programming conventions that you take for granted — the format of each function’s parameter should be understandable to the typical user.

Required code files

There are three API code files that you must use without alteration with your external function plug-ins: FMFlags.h, FMExtern.h, and FMExtern.c. These files are FileMaker Proprietary API files and cannot be distributed in source form without written consent from FileMaker, Inc.

FMFlags.h

The FMFlags.h header file contains compiler directives to control code compilation. This file allows you to have one set of source code files that will compile on Windows and Mac OS machines. Do not alter this file.

FMExtern.h and FMExtern.c

The FMExtern.h is the header file for the FMExtern.c file. Together, these files define the parameter block (the shared data structure used by your plug-in and FileMaker Pro) and some shared function calls. The function calls are used to manipulate the parameter and result handles in the parameter block.

The FMExtern.h file defines the call-back functions for memory operations and the different kinds of plug-in events (FileMaker Pro messages) sent to the plug-in in a FMExternCallSwitch definition.

The FMExtern.c file contains a 68K callback mechanism to handle Power PC applications. Do not alter these files.

```
typedef enum { kFMXT_Init, kFMXT_Idle, kFMXT_Internal1,
              kFMXT_External, kFMXT_Shutdown,
              kFMXT_DoAppPreferences, kFMXT_Internal2 }
FMExternCallSwitch;
```

The kFMXT_Internal1 and kFMXT_Internal2 values are reserved by FileMaker, Inc. For information about the other values, see “FileMaker Pro messages sent to the plug-in” on page 12-10.

FMExternCallStruct defines the structure of the parameter block. FMExternCallPtr is a pointer to that structure and gFMExternCallPtr is a global variable that is defined as an FMExternCallPtr pointer.

Within the FMExternCallStruct definition are three variables of type long: param2, param3 and result. The param2 variable contains the ID (0, 1, 2, etc.) for the external function referenced from the Specify Calculation dialog box in FileMaker Pro.

The param3 variable contains the value of the expression that replaces the external function’s parameter passed to the plug-in from the calculation formula in FileMaker Pro. Since external functions can only return text/string data, the data in the param3 variable and the data you put into the result variable must be text or a textual representation of a number.

Note Param3 is a *pointer* (type long integer) to a text variable.

Required resource files

Two resource files are provided, that define the string resources required by FileMaker Pro. The resource files are FMExample.rc for Windows, and FMExample.r for Mac OS. For Windows, the string resources start at string ID 128 in the resource file. For Mac OS, they’re in a STR# resource with ID 128 in the resource file.

These four string resources are required for a FileMaker Pro plug-in:

- The first string (ID 128) is the plug-in’s name as it appears in the FileMaker Pro Application Preferences dialog box.

- The second string (ID 129) is the descriptive text displayed in the Application Preferences dialog box when the plug-in is selected.
- The third string (ID 130) must be empty.
- The fourth string (ID 131), referred to as the *feature string*, contains the plug-in's unique ID and feature flags.

```

////////////////////////////////////
//
// String Table
//

STRINGTABLE DISCARDABLE
BEGIN
    128 "FMExample" /* Name of plugin as shown in calculation
dialog */
    129 "This is an example plug-in that demonstrates how to
write plug-ins. This comes with FileMaker Developer." /* De-
scription shown in application preferences panel in FileMaker
*/
    131 "Xmpl1YYYYnY" /* "Feature String" = <4 char creator>
<always 1> <app prefs> <has external functions> <always Y>
<idle> <always n> <Win32s> */

    144 "Xpl-Version" /* functionId 0 */
    145 "Xpl-NumToChar" /* functionId 1 */
    146 "Xpl-CharToNum" /* functionId 2 */
    147 "Xpl-Format" /* functionId 3 */
    148 "Xpl-NumWords" /* functionId 4 */

/* INSERT YOUR CODE HERE */
// 149 "Xpl-MyNewFunctionNameHere" /* functionId 8 */

    kErrorMsg "FMExample Plugin Error"
    kErrorMsg "error %li"
    kStrMsg "FMExample Plugin"
    kStrBody "%s"

END
////////////////////////////////////
////////////////////////////////////

```

String table in FMExample.rc

```

resource 'STR#' (kFMEX_RES_STRINFOID, purgeable) {
    { /* array StringArray: 4 elements */
        /* [1] */
        /* Name of plugin as shown in calculation dialog */
        "FMExample",
        /* [2] */
        /* Description shown in application preferences
panel in FileMaker */
        "This is an example plug-in that demonstrates how
to write plug-ins. This comes with FileMaker
Developer.",
        /* [3] */
        "",
        /* [4] */
        /* "Feature String" = <4 char creator> <always 1>
<app prefs> <has external functions> <always Y>
<idle> <always n> <Win32s> */
        "Xmpl1YYYYnY"
    }
}; /* STR# kFMEX_RES_STRINFOID */

```

String table in FMExample.r

Feature string syntax

The feature string must be 11 characters long for FileMaker Pro plug-ins.

The first four characters of the feature string are the ID of the FileMaker Pro plug-in. The ID must be unique for each plug-in and must not begin with "F," "FM," or "Web." For the Mac OS, it is recommended that you set the creator type of the plug-in to this same value. The ID can only contain low-ASCII alphanumeric characters (such as 0-9, A-Z, and a-z).

Note So that there will be a good chance of having a unique ID, you should register the ID at the Apple Developer Support web site—even if you won't be creating a Mac OS version of your plug-in. To register plug-in IDs, go to the developer support pages on the Apple Computer web site at www.apple.com/developer/ (see "Registering your plug-ins" on page 12-13).

The fifth character of the feature string is always “1,” the eighth is always “Y,” and the tenth is always “n.” Other values for these flags are reserved for FileMaker, Inc. use only. Here’s a description of each character in the feature string:

Characters in the feature string	Description of characters
1-4•	Characters 1-4 are the plug-in ID. (Register the ID on the Apple Developer Support web site at www.filemaker.com/developers/index.html .)
5	Character 5 is always “1.”
6•	Set the sixth character of the feature string to “Y” if you want to enable the Configure button for plug-ins in the Application Preferences dialog box. Use “n” if there is no plug-in configuration needed. If the flag is set to “Y,” then make sure to handle the <code>kFMXT_DoAppPreferences</code> message. (See “FileMaker Pro messages sent to the plug-in” on page 12-10.)
7•	Set the seventh character of the feature string to “Y” when the plug-in has external functions you want to appear in the Specify Calculations dialog box. It is expected that this is going to be “Y” most of the time. The names of the external functions must start at string ID 144 (Windows) or be in a “STR#” resource with ID 144 (Mac OS). The first external function name in the list is assigned the ID of zero and is increased by one for each string found after that. (In Windows, FileMaker Pro stops scanning for names when it encounters the first empty string after string ID 144.)
8	Character 8 is always “Y.”

Characters in the feature string

Description of characters

9•	Set the ninth character of the feature string to “Y” if the <code>kFMXT_Idle</code> message is required. For simple external functions this may not be needed and can be turned off by setting the character to “n.”
10	Character 10 is always “n.”
11•	Set the eleventh character to “n” if the plug-in cannot limit its Windows calls to the Win32s subset. (This might be the case for FileMaker Pro 4.0 plug-ins.) Otherwise, it should be set to “Y.” Even though the state of the Win32s flag is ignored in the Mac OS, it still must exist.

For example, “Moc31nYYnnY” is a feature string for a plug-in with the ID of “Moc3” (characters 1-4) that includes an external function (character 7 = “Y”), does not need any special configuration or idle time (characters 6 and 9 both = “n”), and uses only the Win32s API (character 11 = “Y”).

Requirements for the plug-in’s main entry point

The `FMMain.h` and `FMMain.c` files are examples of what the main file should look like for a FileMaker Pro plug-in. They define the main entry point function for the various environments (Windows and Mac OS) that a plug-in can be compiled for.

The main entry point contains a switch statement that allows the plug-in to determine the condition under which control is passed to it, so that it may act appropriately.

```
switch (pb->whichCall) {
```

The variable “pb” is a local variable that was assigned to be equal to `gFMExternCallPtr`. (See the description of the `FMExtern.h` file in “Required code files” on page 12-7).

Main entry point for Windows environments

The Windows platform plug-in exists as a 32-bit DLL with one exported, named entry point. The filename extension for the DLL must be “.FMX” and the name of the entry point must be “FMExternCallProc.”

There should be no need for any custom DLLMain routine because the instance Handle of the DLL will be passed each time the DLL is called.

Main entry point for Mac OS environments

The Mac OS platform plug-in can contain any combination of PowerPC and 680x0 based code. If the plug-in requires a PowerPC processor to run, you must still write a small 680x0 code resource. This code resource should cause the plug-in to refuse to load during initialization after displaying an appropriate message.

680x0 computers The 680x0 code exists as a resource of type FMXT with an ID of 1024 and the entry point is at the first byte of that resource. Because FileMaker Pro 4.0 runs on 680x0 machines, make sure that at least the entry point and the initialization handler are not compiled for 68020 or later computers if your plug-in is for FileMaker Pro 4.0.

PowerPC computers The PowerPC code exists as a shared library in the data fork of the plug-in with a single exported named entry point. The name of the entry point is FMExternCallProc and the file system's file type for the plug-in must be FMXT. (In Code Warrior Pro, set all the fields of the PPC PEF preferences panel to 1024. The PEF's fragment name is ignored.)

External function naming conventions

The function name prefix for all of the plug-in's external functions must be a unique value containing 4 or 5 characters and must not begin with the characters “FM” or “Web.” Three-character prefixes are reserved by FileMaker, Inc.—for example, the FMExample plug-in's function name prefix is “Xpl.”

FileMaker, Inc. will manage the naming conventions for plug-in name, filename, and function prefix. For this reason, you need to register your plug-in.

In addition, FileMaker, Inc. has reserved certain naming conventions for external functions—for example, the FileMaker Pro Web Companion uses the naming convention Web-XXXX, where XXXX is the name of the specific Web Companion external function.

See “Registering your plug-ins” on page 12-13 for more information.

FileMaker Pro messages sent to the plug-in

There are five possible calls that FileMaker Pro can request of your plug-in. Messages sent by FileMaker Pro to your plug-in are supplied in the whichCall field of the parameter block, FMExternCallStruct, defined in the FMExtern.h file.

- kFMXT_Init — the Initialization message
- kFMXT_Shutdown — the Shutdown message
- kFMXT_Idle — the Idle message
- kFMXT_DoAppPreferences — the Preferences message
- kFMXT_External — the External message

The Initialization message

The Initialization message, `kFMXT_Init`, is sent to the plug-in whenever it is enabled in FileMaker Pro. This may or may not correspond with the startup of the FileMaker Pro application, depending on whether the plug-in is enabled in the Application Preferences dialog box.

There are three possible result values that the plug-in should return in response to the Initialization message:

- `kBadExtnVersion` should be returned if the version number passed is less than the value of `kMinExtnVersion` or greater than the value of `kMaxExtnVersion`. This prevents the plug-in from running on an incompatible version of the API it was compiled with.
- `kDoNotEnable` should be returned if the plug-in should not be enabled. This could be the result of a specific piece of hardware missing or that a certain system software component was not of the proper version number. If the plug-in is going to return this value, it should first display some type of alert telling the user why it could not be enabled.
- `kCurrentExtnVersion` is the only other result value that should be returned. This causes the plug-in to be enabled.

For the `FMExample` plug-ins, the `Do_PluginInit` function is called when the Initialization message is received. The `Do_PluginInit` function first checks the version of the API that the plug-in was compiled with to verify if it's compatible with the version of FileMaker Pro that has loaded it. Then the function checks for preferences and sets them if they exist. If no preferences currently exist, it will create them with default values.

In Windows, these preferences are stored as registry entries. In the Mac OS, they are stored in a file within the Preferences folder of the System Folder. Due to the differences between the way this information is stored on the two platforms, the `Do_PluginInit` function uses preprocessor instructions to choose the correct code at compile time. The preprocessor flags controlling the selection process can be found in the `FMFlags.h` file.

If the preferences are set properly and the API version is okay, the `Do_PluginInit` function in the `FMExample` plug-in will return `kCurrentExtnVersion`—otherwise, it will return an error and the plug-in will not be enabled by FileMaker Pro.

The Shutdown message

The Shutdown message `kFMXT_Shutdown`, is sent to the plug-in whenever it is disabled in FileMaker Pro. This may or may not correspond with the quitting of the FileMaker Pro application, depending on whether the plug-in is disabled in the Application Preferences dialog box.

The `FMExample` plug-in does not allocate any persistent memory on the heap, and therefore does not do anything when it receives the Shutdown message. You should implement a clean-up function in your plug-in, however, to deallocate anything you have on the heap and exit from any OS services you may be using. Remember that it's possible for a plug-in to be enabled and disabled multiple times during a session so it's important that your plug-in will clean up memory.

The Idle message

The Idle message, `kFMXT_Idle`, is only sent to the plug-in during idle time if the idle feature flag was set to “Y” in the feature string and the plug-in is currently enabled.

There are two times when this message is called by the FileMaker Pro application:

- If the `unsafeCalls` parameter is non-zero, then the routine has been called at the same time that the FileMaker Pro application's low-level networking code has been called.

Do not perform any lengthy, user interface, or event processing when the `unsafeCalls` parameter is non-zero.

- The other time when the Idle message will be sent is when FileMaker Pro detects free time and does its own internal idle handling.

The External Function message

The External Function message, `kFMXT_External`, is sent to the plug-in when FileMaker Pro is processing a calculation that contains one of the plug-in's external functions. This is where the majority of the action takes place.

The names of the external functions start at string ID 144 for Windows and are defined as a `STR#` resource with ID 144 for the Mac OS. The first exported function name is mapped to ID 0 and increased incrementally by 1 for each following name. These ID numbers are then used to switch to the correct processing routine.

The input and result parameters of the External Function message are Mac OS style Handles that contain text using Macintosh character set encoding. This means that all numbers are also represented as text. The result Handle is always empty when the External Function message is sent. You should only manipulate the Handles using the memory manager callback routines that are defined in `FMExtern.h`.

In the Mac OS, the parameters will be real Mac OS Handles, but you cannot depend on which heap they may be located in. In Windows, these Handles exist in the FileMaker Pro memory manager pool and have no relation to anything called a `HANDLE` in the Win32s APIs.

Depending on the value of the `param2` variable, the `FMExample` plug-in (`FMFunc.c`) will run the following external functions:

This param2 value	Corresponds to this external function name	And causes the plug-in to run this function
0	Xpl-Version	PlugInVersion
1	Xpl-NumToChar	SimpleNumToChar
2	Xpl-CharToNum	SimpleCharToNum
3	Xpl-Format	funcnt_Format
4	Xpl-NumWords	funcnt_Num2Words

For information on each external function, see "Description of the `FMExample` plug-in's external functions" on page 12-4.

The Preferences message

The Preferences message, `kFMXT_DoAppPreferences`, is sent in response to a user clicking the **Configure** button for the selected plug-in in the Application Preferences dialog box.

The plug-in should display a dialog box of some sort that will allow the user to set any specific configuration data required by the plug-in. If the plug-in requires user-definable preferences, you should implement your UI here. The **Configure** button will only be enabled if the sixth character of the feature string is set to "Y" (see "Feature string syntax" on page 12-8).

Any options that need to be saved should be placed in their own registry entry or `.INI` file (Windows) or in their own preference file (Mac OS).

The `FMExample` plug-in needs to implement a configuration dialog box for the `Xpl-Format` function, so the flag has been set in the feature string (`Xmpl1YYYnnn`) and the function `Do_PluginPrefs` is called when the Preferences message is received.

Debugging your plug-in

Your plug-in code could include a preprocessor instruction to cause your plug-in to behave differently during the debugging process. For example, the FMExample plug-in will behave in the following ways when the `DEBUG_VERSION` flag is set to “1”:

- The `DebugPlugin` routine will be substituted for the `PluginVersion` function.
- When a message for `Xpl-Version` is received, the routine returns a debug string instead of the version or platform string.
- `DebugPlugin` will return different debug strings depending on the value of the parameter that is passed in.

Avoiding potential Mac OS resource conflicts

Problems can occur on Mac OS machines if your plug-in has the same ID for a resource that FileMaker Pro or another plug-in has for the same type of resource. To avoid potential resource ID conflicts with your plug-in and other applications or plug-ins, follow these guidelines:

- **Use ID numbers between 23,000 and 24,999**

Use hard-coded IDs from this range for your dialog boxes, sounds, icons, and other resources to avoid conflicts with FileMaker Pro resources. FileMaker Pro does not use any of the IDs in this range for the application resources.

- **Set the current resource file to your plug-in**

To avoid conflicts with other plug-ins that use the same resource IDs, use the Mac OS toolbox call in the Resource Manager to set the current resource file to your plug-in before getting any resource objects from the resource file.

Include the following line before any line that references or uses a resource:

```
UseResFile (pb -> resourceID) ;
```

When FileMaker Pro loads your plug-in, the application gives the resource ID. This is located in the parameter block near the `param2` and `param3` variables in the `FMExtern.h` file (see “Required code files” on page 12-7).

Providing documentation for your plug-in

Your FileMaker Pro plug-in should include an example database file with any special fields and scripts necessary to demonstrate the use of the plug-in’s external functions. In addition, you should provide documentation that describes each external function and its parameters.

For ideas on how to document your plug-in, see other external function plug-ins registered with FileMaker Pro (at www.filemaker.com).

Registering your plug-ins

Register your external function plug-in with FileMaker, Inc. to ensure that it’s unique and not in use by any other plug-in. Registering also allows you to make your plug-in visible to customers searching for a FileMaker Pro plug-in that might suit their needs.

Before registering your plug-in, you can search to see if the plug-in name or feature string ID you are requesting has already been assigned.

Steps for registering your external function plug-in

1. In your web browser, go to www.filemaker.com, click the Support tab, and click the Plug-ins link to display the Plug-ins search page.

2. Click the **Registering a Plug-in** link to display the online registration form.

3. Enter information about you, your company, and others who can be contacted about your plug-in.

4. Enter the required information about your plug-in. The following table describes some of the required fields:

Plug-in Name •	Your plug-in's name as it will appear in the FileMaker Pro Application Preferences dialog box. The plug-in name "Web Companion" is already registered and any names that begin with the letters "FM" are reserved.
Plug-in Filename•	The filename(s) of your plug-in for Windows and/or Mac OS platforms. The filename "Web Companion" is already registered and any filenames that begin with the letters "FM" are reserved.
Feature String ID•	The plug-in's feature string ID. All feature string IDs that begin with the letter "F" are reserved. See "Feature string syntax" on page 12-8.
Function Name Prefix•	The prefix that identifies every external function for the plug-in. The prefix must be 4 to 5 characters long. All three-character prefixes and any prefixes beginning with the letters "FM" are reserved by FileMaker, Inc.

5. Fill out the rest of the form as applicable and then click **Submit Plug-in**.

You will be informed during online registration if the plug-in name or feature string ID you requested is already assigned and you will be asked to submit an alternate name or ID. If the name and ID are available, your registration will be accepted and you will receive a temporary online registration confirmation.

Currently, the registration system will only hold plug-in names and feature string IDs for six months. After that time, the registration system automatically deletes the names and IDs unless FileMaker, Inc. receives your actual external function plug-in.

6. Send/upload a copy of your external function plug-in to FileMaker, Inc. as outlined in the online registration confirmation.

FileMaker, Inc. Technical Support must have access to all registered plug-ins. Once FileMaker, Inc. receives both your online registration and your external function plug-in, an official confirmation will be sent to you via email.

7. Repeat these steps to register additional plug-ins.

You must submit a separate registration form for each external function plug-in.

Revising a registered plug-in

If you need to revise the information about an external function plug-in that is already registered to you, you must send an email message to FileMaker, Inc. at plugins@filemaker.com. Please be sure to provide the following information:

- The registration ID number that was assigned to you when you first registered your plug-in.
- Your name.
- Your full company name.
- Your daytime phone number.
- The name of the product with registered information you want to revise.
- Include any changes you want to make.
- If applicable, send the revised plug-in file.

A confirmation of the revision will be sent to you.

Appendix A

Feature comparison of the runtime application and FileMaker Pro

When you double-click the FileMaker Developer application icon to start the application, the New Database dialog box opens automatically, from which you can open a database file. When you start a FileMaker Pro runtime application, the primary bound database file opens automatically.

Other key differences between the runtime application and FileMaker Pro include the following:

- All the database design features have been removed or hidden in the runtime application.

These include the Define Fields, Define Relationships, Define Value Lists, Layout Mode, ScriptMaker, and Access Privileges menu commands.

- Some other menu commands have been removed from the runtime application.

For example, you can't use the runtime application to create, open, or close a database. (Bound runtime database files must contain a custom button or script to close or open other files. There is no close box on a runtime database window.)

- The Scripts menu can be named something different in the runtime application.
- FileMaker Pro Help is not available in the runtime application. However, the Help menu and the Apple menu can contain custom About and Help menu commands.
- Some tools are unavailable on the toolbars in Browse mode, Find mode, and Preview mode in the runtime application. Toolbars in runtime solutions are not supported in Mac OS X.

- The runtime application doesn't support any FileMaker Pro Companions. Features that use the Web Companion aren't available. Also, you can't use the Data Access Companion (which allows ODBC connectivity) with the runtime application.

FileMaker Pro File Sharing, serving a database on the Web, or communicating with a Java applet requires FileMaker Pro. You can, however, use FileMaker Server 5.x to serve runtime applications.

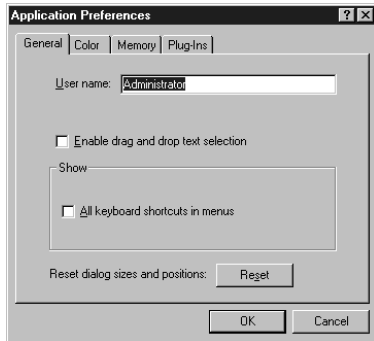
Although the Web Companion, Local Data Access Companion, and Remote Data Access Companion can't be accessed by the runtime application, external function plug-ins can be enabled in the Application Preferences dialog box.

- Apple events are supported but OLE automation is not supported in the runtime application on Windows machines.

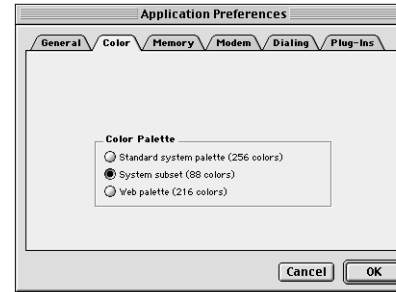
Application and document preferences

In the runtime application, the following options aren't available on the General tab of the Application Preferences dialog box:

- Show templates in New Database dialog check box
- Show recently opened files check box
- Network protocol options

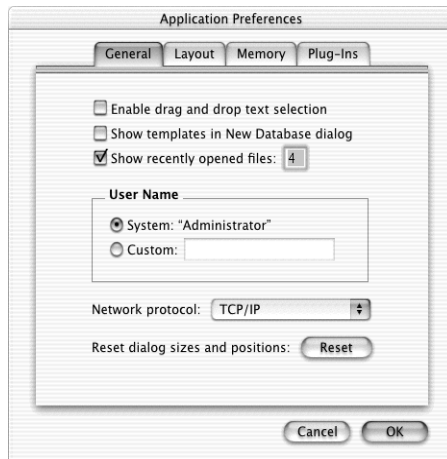


General preferences in the runtime application (Windows)

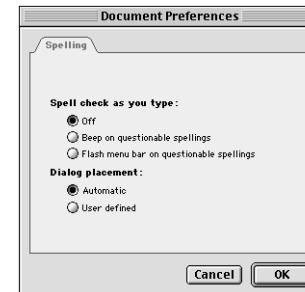


Color application preferences in the runtime application

The Document Preferences dialog box does not have the General tab in the runtime application, only the Spelling tab.

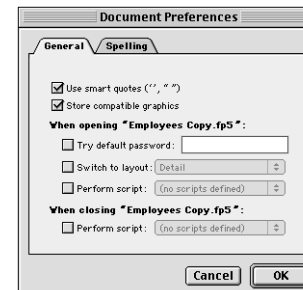


General preferences in the runtime application (Mac OS X)



Spelling document preferences in the runtime application

The Layout tab is changed to the Color tab in the Application Preferences dialog box for the runtime application.



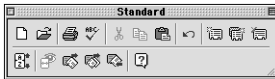
General and spelling document preferences in FileMaker Pro

Toolbar comparison

The New Database and Open tools in the standard toolbar (in Browse mode, Find mode, and Preview mode) are not available in the runtime application.

The Help tool on the standard toolbar in the runtime application is dimmed unless a custom Help script has been specified.

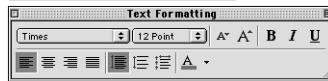
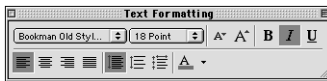
The text formatting toolbar is the same for both the runtime application and FileMaker Pro.



Browse mode in FileMaker Pro



Browse mode in the runtime application



Menu command comparison

The following table shows the menu commands that are available in FileMaker Pro and in the runtime application.

Note In the runtime application, the Format menu is unavailable in Preview mode.

File Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
New Database	•		•		•	
Open	•		•		•	
Open Remote	•		•		•	
Close	•		•		•	
Define Fields	•		•		•	

File Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
Define Value Lists	•		•		•	
Define Relationships	•		•		•	
Database Design Report	•		•		•	
Access Privileges	•		•		•	
Change Password		•		•		•
Sharing	•		•		•	
Print Setup	•	•				
Page Setup			•	•	•	•
Print	•	•	•	•	•	•
Import Records	•	•	•	•	•	•
Export Records	•	•	•	•	•	•
Save a Copy As	•	•	•	•	•	•
Recover	•	1	•	2	•	2
<List of previously opened files>	•		•		•	
Exit	•	•				
Quit			•	•		

¹ Press Ctrl+Shift

² Press Option+⌘

Edit Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
Undo	•	•	•	•	•	•
Cut	•	•	•	•	•	•
Copy	•	•	•	•	•	•
Paste	•	•	•	•	•	•
Paste Special	•	•				

Edit Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
Clear	•	•	•	•	•	•
Select All	•	•	•	•	•	•
Find /Replace	•	•	•	•	•	•
Spelling	•	•	•	•	•	•
Object	•	•				
Preferences	•	•	•	•	1	1

¹ See Application Menu table

View Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
Browse Mode	•	•	•	•	•	•
Find Mode	•	•	•	•	•	•
Layout Mode	•		•		•	
Preview Mode	•	•	•	•	•	•
View as Form	•	•	•	•	•	•
View as List	•	•	•	•	•	•
View as Table	•	•	•	•	•	•
Toolbars	•	•	•	•		
Status Bar	•	•				
Status Area	•	•	•	•	•	•
Text Ruler	•	•	•	•	•	•
Zoom In	•	•	•	•	•	•
Zoom Out	•	•	•	•	•	•

Insert Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
Object	•	•				
Picture	•	•	•	•	•	•
QuickTime	•	•	•	•	•	•
Sound	•	•	•	•	•	•
Current Date	•	•	•	•	•	•
Current Time	•	•	•	•	•	•
Current User Name	•	•	•	•	•	•
From Index	•	•	•	•	•	•
From Last Record	•	•	•	•	•	•

Format Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
Font	•	•	•	•	•	•
Size	•	•	•	•	•	•
Style	•	•	•	•	•	•
Align Text	•	•	•	•	•	•
Line Spacing	•	•	•	•	•	•
Text Color	•	•	•	•	•	•
Text	•	•	•	•	•	•

Records Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
New Record	•	•	•	•	•	•

Records Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
Duplicate Record	•	•	•	•	•	•
Delete Record	•	•	•	•	•	•
Delete All Records	•	•	•	•	•	•
Modify Last Find	•	•	•	•	•	•
Show All Records	•	•	•	•	•	•
Show Omitted	•	•	•	•	•	•
Omit Record	•	•	•	•	•	•
Omit Multiple	•	•	•	•	•	•
Sort	•	•	•	•	•	•
Replace Contents	•	•	•	•	•	•
Relookup Contents	•	•	•	•	•	•
Revert Record	•	•	•	•	•	•

Requests Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
(Find mode)						
Add New Request	•	•	•	•	•	•
Duplicate Request	•	•	•	•	•	•
Delete Request	•	•	•	•	•	•
Show All Records	•	•	•	•	•	•
Perform Find	•	•	•	•	•	•
Constrain Found Set	•	•	•	•	•	•
Extend Found Set	•	•	•	•	•	•
Revert Request	•	•	•	•	•	•

Scripts Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
ScriptMaker...	•		•		•	
Debug Scripts	•		•		•	
<Script names>	•	•	•	•	•	•

Window Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
Tile Horizontally	•	•				
Tile Vertically	•	•				
Cascade	•	•				
Arrange Icons	•	•				
Hide Window			•	•	•	•
Minimize Window					•	•
Bring all to front					•	•
<Names of open files>	•	•	•	•	•	•

Help Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
FileMaker Pro Help	•		•		•	
Contents and Index	•		•		•	
What's This?	•					
FileMaker on the Web	•		•		•	
Send Feedback to FileMaker	•		•		•	
Register Now	•		•		•	
About FileMaker Pro	•		1		2	
About Balloon Help			•	•		
Show Balloons			•	•		
About FileMaker Pro Runtime (Displays if no custom About script is specified)		•		1		2
About <runtime solution> (Displays if custom About script is specified)		•		1		2

Help Menu command	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
<Runtime solution Help script name> (Displays if custom Help script is specified)		•		•		•

¹ See Apple Menu command table

² See Application Menu command table

Apple Menu command (Mac OS 9 only)	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
About FileMaker Pro Runtime (Displays if no custom About script is specified)				•		
About <runtime solution> (Displays if custom About script is specified)				•		
About FileMaker Pro			•			

Application Menu command (Mac OS X only)	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
About FileMaker Pro					•	
About FileMaker Pro Runtime (Displays if no custom About script is specified)						•
About <runtime solution> (Displays if custom About script is specified)						•
Preferences					•	•
Services					•	•
Hide FileMaker Pro					•	
Hide <runtime solution>						•

Application Menu command (Mac OS X only)	Windows		Mac OS 9		Mac OS X	
	Pro	RT	Pro	RT	Pro	RT
Hide Others					•	•
Show All					•	•
Quit FileMaker Pro					•	
Quit <runtime solution>						•

Ignored script steps

Because some features have been removed from the runtime application, the following script steps are ignored by the runtime application.

- Print Script Definitions (instead, the current record is printed)
- Print Field Definitions (instead, the current record is printed)
- Open Define Fields
- Open Define Relationships
- Open Define Value List
- Open Sharing
- Open ScriptMaker
- Open Help (executes custom Help script specified during binding)
- Set Multi-User
- New Database
- Perform External Script (if the specified file has not been bound to the runtime application)
- Open File (if the specified file has not been bound to the runtime application)

Stored registry settings

On Windows machines, FileMaker Pro stores its registry settings at `hkey_current_user\software\FileMaker\FileMaker Pro\6.0D`

The runtime application stores its registry settings at `hkey_current_user\software\FileMaker\<solution name>\6`

Note The filename extension for the runtime database files is registered at `hkey_classes_root`.

On Mac OS machines, the Developer edition of FileMaker Pro stores its preferences in the FileMaker Pro 6.0D Prefs file inside the FileMaker Preferences folder. The runtime application stores its registry settings in the <Solution name> 6 Prefs file inside the FileMaker Preferences folder.

On Mac OS X machines, the files are the same except that the FileMaker Preferences folder is in the <Home>\Library\Preferences\ folder.

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix B

Valid names used in CGI requests for FileMaker Pro XML data

This appendix describes the valid names of requests and their parameters you can use in a CGI (Common Gateway Interface) command when requesting FileMaker Pro data in XML format. Included are HREF link and HTML form examples of each request. For scripting examples of each request, see the JavaScript library, FMP.js, in the XML Examples folder.

The CGI command requesting FileMaker Pro data in XML format always begins with the action as FMPPro? and the format file as either -dso_xml, -dso_xml_dtd, -fmp_xml, or -fmp_xml_dtd.

The following is a list of the request names and parameters:

Request names	Parameter names
-find	-db
-findall	-lay
-findany	-format
-view	-recid
-new	-modid
-edit	-lop, -op
-delete	-max
-dbnames	-skip
-layoutnames	-sortorder, -sortfield
-scriptnames	-script, -script.prefind, -script.presort
-dbopen	-styletype, -stylehref
-dbclose	-password
-dup	
-img	

Note The name of a database field can also be used in a CGI command. It is not a parameter or a request name, and therefore is not preceded by a hyphen (-).

Generating a -find, -findall, or -findany request

Name/Value Type: Find Record(s) Request

What it does: Submits a search request using defined criteria.

A web user must have browsing access privileges with the database in order to execute these requests.

Required parameters: -db, -format

Optional parameters: -lay, -recid, -lop, -op, -max, -skip, -sortorder, -sortfield, -script, -script.prefind, -script.presort, field name, -styletype, -stylehref

Examples of -find, -findall, and -findany requests

To find a record using a hypertext link:

```
<a href="FMPPro?-db=employees.fp5&-format=-fmp_xml &Country=USA&-max=1&-find">Find first USA record</a>
```

To find all records in the database using a hypertext link:

```
<a href="FMPPro?-db=employees.fp5&-format=-fmp_xml &-findall">Find all records</a>
```

To find any record using a hypertext link:

```
<a href="FMPro?-db=employees.fp5&-format=-fmp_xml &-findany">Find a random record for today's daily quote</a>
```

To find some records using a form action:

```
<form action="FMPro" method="post">
<input type="hidden" name="-db" value="employees.fp5">
<input type="hidden" name="-format" value="-fmp_xml">
<input type="hidden" name="-max" value="all">
<input type="text" size=12 name="Country" value="USA">
<input type="submit" name="-find" value="Find Records">
</form>
```

Generating a –view request

Name/Value Type: View Request

What it does: Retrieves layout information from a database and displays it in the FMPXMLLAYOUT grammar.

Required parameters: –db, –lay, –format (= –fmp_xml)

Optional parameters: –styletype, –stylehref

Examples of –view requests

To retrieve layout information using a hypertext link:

```
<a href="FMPro?-db=employees.fp5&-lay=LayoutOne&-format=-fmp_xml&-view">"Take me to a search page"</a>
```

To retrieve layout information using a form action:

```
<form action="FMPro" method="post">
<input type="hidden" name="-db" value="employees.fp5">
<input type="hidden" name="-lay" value="LayoutOne">
<input type="hidden" name="-format" value="-fmp_xml">
```

```
<input type="submit" name="-view" value="Show Search Page">
</form>
```

Generating a –new request

Name/Value Type: New Record Request

What it does: Creates a new record and populates that record with the contents of any field name/value pairs.

A web user must have access privileges for creating records in order to execute this request.

Required parameters: –db, –format, one or more field name(s)

Optional parameters: –styletype, –stylehref

Note To include new data for a portal, you must also specify the layout and the relationship name for the related database followed by two colons. You can only add one row/record to a portal per request. The –max parameter returns 0 if the request returns no records.

Examples of –new requests

To create a new record using a hypertext link:

```
<a href="FMPro?-db=employees.fp5&Country=Australia&-format=-fmp_xml&-new">
```

To create a new record using a form action:

```
<form action="FMPro" method="post">
<input type="hidden" name="-db" value="employees.fp5">
<input type="hidden" name="-format" value="-fmp_xml">
<input type="text" size=12 name="Country" value="Australia">
<input type="submit" name="-new" value="New Record">
</form>
```

Generating an *–edit* request

Name/Value Type: Edit Record Request

What it does: Updates the record specified by *–recid*, populating the fields with the contents of any field name/value pairs.

The *–recid* parameter indicates which record should be edited. In order to edit a record, the web user must have editing privileges for the database.

Required parameters: *–db*, *–format*, *–recid*, one or more field name(s)

Optional parameters: *–modid*, *–styletype*, *–stylehref*

Note To edit records in a portal, you must also include the layout name and the relationship name followed by two colons. To specify each record in the portal, include a period and a consecutive number after the related field name, such as *address::city.1* for the first row (record) in the portal and *address::city.2* for the second. (See “Requests for editing multiple records in a portal” on page 10-9 for information.)

Examples of *–edit* requests

To edit a record using a hypertext link:

```
<a href="FMPro?–db=employees.fp5&–format=–fmp_xml&–recid=13&Country=USA&–edit">
```

To edit a record using a form action:

```
<form action="FMPro" method="post">
<input type="hidden" name="–db" value="employees.fp5">
<input type="hidden" name="–format" value="–fmp_xml">
<input type="hidden" name="–recid" value="13">
<input type="text" size=12 name="Country" value="Type a country name here">
<input type="submit" name="–edit" value="Edit This Record">
</form>
```

Generating a *–delete* request

Name/Value Type: Delete Record Request

What it does: Deletes the record as specified by *–recid* parameter. In order to delete a record, the web user must have record deleting privileges for the database.

Required parameters: *–db*, *–format*, *–recid*

Optional parameters: *–styletype*, *–stylehref*

Examples of *–delete* requests

To delete a record using a hypertext link:

```
<a href="FMPro?–db=employees.fp5&–format=–fmp_xml&–recid=4&–delete">Delete record with ID 4</a>
```

To delete a record using a form action:

```
<form action="FMPro" method="post">
<input type="hidden" name="–db" value="employees.fp5">
<input type="hidden" name="–format" value="–fmp_xml">
<input type="hidden" name="–recid" value="4">
<input type="submit" name="–delete" value="Delete This Record">
</form>
```

Generating a *–dbnames* request

Name/Value Type: Database Names Request

What it does: Retrieves the names of all databases that are open and shared via the Web Companion.

Required parameters: *–format*

Optional parameters: *–styletype*, *–stylehref*

Examples of *-dbnames* requests

To retrieve the database names using a hypertext link:

```
<a href="FMPro?-dbnames &-format=-fmp_xml&-styletype=
text/css&-stylehref=mystylesheet.css">
```

To retrieve the database names using a form:

```
<form action="FMPro" method="post">
<input type="hidden" name="-format" value="-fmp_xml">
<input type="hidden" name="-styletype" value="text/css">
<input type="hidden" name="-stylehref" value="mystylesheet.css">
<input type="submit" name="-dbnames" value="Show Database List">
</form>
```

Generating a *-layoutnames* request

Name/Value Type: Layout Names Request

What it does: Retrieves the names of all available layouts for a specified database that is open and shared via the Web Companion.

Required parameters: *-db*, *-format*

Optional parameters: *-styletype*, *-stylehref*

Examples of *-layoutnames* requests

To retrieve the names of available layouts using a link:

```
<a href="FMPro?-db=employees.fp5&-layoutnames&-format=
-fmp_xml&-styletype=text/css&-stylehref=mystylesheet.css">
```

To retrieve the names of available layouts using a form:

```
<form action="FMPro" method="post">
<input type="hidden" name="-db" value="employees.fp5">
<input type="hidden" name="-format" value="-fmp_xml">
<input type="hidden" name="-styletype" value="text/css">
```

```
<input type="hidden" name="-stylehref" value="mystylesheet.css">
<input type="submit" name="-layoutnames" value="Show List of
Layouts">
</form>
```

Generating a *-scriptnames* request

Name/Value Type: Script Names Request

What it does: Retrieves the names of all available scripts for a specified database that is open and shared via the Web Companion.

Required parameters: *-db*, *-format*

Optional parameters: *-styletype*, *-stylehref*

Examples of *-scriptnames* requests

To retrieve the names of all scripts using a link:

```
<a href="FMPro?-db=employees.fp5&-scriptnames&-format=
-fmp_xml&-styletype=text/css&-stylehref=mystylesheet.css">
```

To retrieve the names of all scripts using a form:

```
<form action="FMPro" method="post">
<input type="hidden" name="-db" value="employees.fp5">
<input type="hidden" name="" value="">
<input type="hidden" name="-format" value="-fmp_xml">
<input type="hidden" name="-styletype" value="text/css">
<input type="hidden" name="-stylehref" value="mystylesheet.css">
<input type="submit" name="-scriptnames" value="Show List of Scripts">
</form>
```

Generating a *–dbopen* request

Name/Value Type: Open Database Request

What it does: Opens a specified database that’s located in the Web folder with Remote Administration enabled in the Web Companion (web users must enter “Admin” as the user name)

Required parameters: *–db*, *–format*

Optional parameter: *–password*

Examples of *–dbopen* requests

To open a remotely administered database using a link:

```
<a href="FMPro?–db=employees.fp5&–dbopen&–format=–fmp_xml&–styletype=text/css&–stylehref=mystylesheet.css">
```

To open a remotely administered database using a form:

```
<form action="FMPro" method="post">
<input type="hidden" name="–db" value="employees.fp5">
<input type="hidden" name="–format" value="–fmp_xml">
<input type="hidden" name="–styletype" value="text/css">
<input type="hidden" name="–stylehref" value="mystylesheet.css">
<input type="submit" name="–dbopen" value="Open Employees
Database">
</form>
```

For more information, see “Opening password-protected databases remotely” on page 8-18.

Generating a *–dbclose* request

Name/Value Type: Close Database Request

What it does: Closes an open database that’s located in the Web folder with Remote Administration enabled in the Web Companion (web users must enter “Admin” as the user name)

Required parameters: *–db*, *–format*

Examples of *–dbclose* requests

To close a remotely administered database using a link:

```
<a href="FMPro?–db=employees.fp5&–dbclose=&–format=–fmp_xml&–styletype=text/css&–stylehref=mystylesheet.css">
```

To close a remotely administered database using a form:

```
<form action="FMPro" method="post">
<input type="hidden" name="–db" value="employees.fp5">
<input type="hidden" name="–format" value="–fmp_xml">
<input type="hidden" name="–styletype" value="text/css">
<input type="hidden" name="–stylehref" value="mystylesheet.css">
<input type="submit" name="–dbclose" value="Close Employees
Database">
</form>
```

For more information, see “Opening password-protected databases remotely” on page 8-18.

Generating a *–dup* request

Name/Value Type: Duplicate Record Request

What it does: Duplicates the record specified by *–recid*.

Required parameters: *–db*, *–format*, *–recid*

Examples of -dup requests

To duplicate the specified record using a link:

```
<a href="FMPro?-db=employees.fp5&-format=-dso_xml&-recid=14&-dup">Duplicate record with ID 14</a>
```

To duplicate the specified record using a form action:

```
<form action="FMPro" method="post">
<input type="hidden" name="-db" value="employees.fp5">
<input type="hidden" name="-format" value="-dso_xml">
<input type="hidden" name="-recid" value="14">
<input type="submit" name="-dup" value="Duplicate record with ID 14">
```

Generating an -img request

Name/Value Type: Retrieve Image Request

What it does: Retrieves the image specified by -img.

Required parameters: -db, -format, -recid, image fieldname

Examples of -img requests

To retrieve the specified image using a link:

```
<a href="FMPro?-db=employees.fp5&-recid=14&logo=FMI Logo.gif&-img">Retrieve the image from the record with ID 14</a>
```

To duplicate the specified record using a form action:

```
<form action="FMPro" method="post">
<input type="hidden" name="-db" value="employees.fp5">
<input type="hidden" name="-recid" value="14">
<input type="hidden" name="logo=FMI Logo.gif">
<input type="submit" name="-img" value="Retrieve the logo from record ID 14">
```

Specifying parameters for the request

The following are the parameters for requesting FileMaker Pro data in XML format. Some parameters are required to be present in the CGI command along with certain requests; others are optional.

-db (Database)

Name/Value Type: Parameter

What it does: Specifies the database that all processing for the request will refer to.

Value is: Name of the database, including the extension if any. The FileMaker Pro Web Companion uses only the name of the database; do not include any path information. The database must be open in FileMaker Pro.

Required with: All requests except the -dbnames request

Example: FMPro?-db=employees.fp5&-format=-fmp_xml&-find

-lay (Layout)

Name/Value Type: Parameter

What it does: Specifies the layout that is used in the database. The -lay parameter is used in -find, -findall, and -findany requests to specify the fields that are returned and in -view requests to specify the layout information that's returned.

Value is: Name of the layout to use. If no layout is given, then the layout is considered to contain all fields in the database (but not related fields).

Required with: -view requests, and -edit or -new requests for data in related fields or portals

Optional with: -find, -findall, or -findany requests

Example: FMPro?-db=employees.fp5&-format=-fmp_xml&-lay=LayoutOne&-view

-format (Format)**Name/Value Type:** Parameter**What it does:** Specifies the XML grammar used for returning the results of a request.**Value is:** `-dso_xml`, `-dso_xml_dtd`, `-fmp_xml`, or `-fmp_xml_dtd`.**Required with:** All requests**Examples:**

To generate this XML grammar	Specify this format
FMPDSORESULT	FMPPro?-db=employees.fp5&-format="-dso_xml"&-find
FMPDSORESULT + document type definition	FMPPro?-db=employees.fp5&-format="-dso_xml_dtd"&-find
FMPXMLRESULT	FMPPro?-db=employees.fp5&-format="-fmp_xml"&-find
FMPXMLRESULT + document type definition	FMPPro?-db=employees.fp5&-format="-fmp_xml_dtd"&-find
FMPXMLLAYOUT	FMPPro?-db=employees.fp5&-format="-fmp_xml"&-view
FMPXMLLAYOUT + document type definition	FMPPro?-db=employees.fp5&-format="-fmp_xml_dtd"&-view

-recid (Record ID)**Name/Value Type:** Parameter**What it does:** Defines which record should be operated on. Used mainly by the `-edit`, and `-delete` requests.**Value is:** A record ID, which is a unique specifier to a record in a FileMaker Pro database.**Required with:** `-edit` and `-delete` requests**Optional with:** `-find` requests**Example:** `FMPPro?-db=employees.fp5&-format=-fmp_xml&-recid=22&-delete`***-modid (Modification ID)*****Name/Value Type:** Parameter**What it does:** Refers to the latest version (incremental counter number) of the record. This allows you to take necessary measures to ensure an `-edit` request is applied to the most current version of the record, by including a warning and an option to retrieve the most current record before the `-edit` request is allowed.**Value is:** A modification ID, which is a unique identifier for the current version of a record in a FileMaker Pro database.**Optional with:** `-edit` requests**Requires:** The `-recid` parameter**Example:** `FMPPro?-db=employees.fp5&-format=-fmp_xml&-recid=22&-modid=6&last_name=Jones&-edit`***-lop (Logical operator)*****Name/Value Type:** Parameter**What it does:** Specifies how the find criteria are combined as either an AND or OR `-find` request.**Value is:** Either AND or OR. If the `-lop` parameter name is not used, then the find request is assumed to be an AND request.**Optional with:** `-find` requests

Example: FMPPro?-db=employees.fp5&-format= -fmp_xml&Last+Name=Smith&Birthdate=2/5/1972&-lop=and&-find

-op (Comparison operator)

Name/Value Type: Parameter

What it does: Specifies the comparison operator to apply to the field name/value pair that follows it in a -find request.

Value is: The operator to use. There are short and long versions of each operator. The default operator is "begins with". Valid operators are as follows:

Short	Long	FileMaker Pro equivalent operator
eq	equals	=word
cn	contains	"word"
bw	begins with	word*
ew	ends with	*word
gt	greater than	> word
gte	greater than or equals	>= word
lt	less than	< word
lte	less than or equals	<= word
neq	not equals	omit, word

You can use any FileMaker Pro -find operator by specifying the begins with (bw) parameter. For example, to specify the "Find Content Match" (=) operator, you would specify the begins with parameter (bw) and then you would place the characters "==" before the actual search criteria. The required lines would look like this:

```
<input type="hidden" name="-op" value="bw">
<input type="text" name="FirstName" value="= Sam">
```

Optional with: -find requests

Requires: A field name and a value

Example: FMPPro?-db=employees.fp5&-format= -fmp_xml&-op=eq&FirstName=Sam&-find

-max (Maximum records)

Name/Value Type: Parameter

What it does: Specifies the maximum number of records that should be returned.

Value is: A number from 1 through 2147483647, or the word "All". The default value is 25.

Optional with: -find requests

Example: FMPPro?-db=employees.fp5&-format= -fmp_xml&-max=10&-find

-skip (Skip records)

Name/Value Type: Parameter

What it does: Tells FileMaker Pro how many records to skip in the found set.

Value is: A number from 0 through 2147483647, or the word "All". If the value is greater than the number of records in the found set or the value is "All" then the last record is displayed. The default value is 0.

Optional with: -find requests

Example: FMPPro?-db=employees.fp5&-format= -fmp_xml&-skip=10&-max=5&-find

In this example, the first 10 records in the found set are skipped and records 11 through 15 are returned.

–sortfield (Sort field)**Name/Value Type:** Parameter

What it does: Specifies the field that will be used for sorting. The `–sortfield` parameter can be used multiple times to perform multiple field sorts. The position in which `–sortfield` appears in the CGI command will determine the sort order of the fields.

Value is: Name of a FileMaker Pro field.**Optional with:** `–find` or `–findall` requests

Example: `FMPPro?–db=employees.fp5&–format= –fmp_xml&–sortfield=First+Name&–find`

–sortorder (Sort order)**Name/Value Type:** Parameter

What it does: Indicates the direction of a sort. If used, `–sortorder` must directly follow the `–sortfield` parameter it applies to.

Value is: The sort order. Valid sort orders are as follows, where Custom is the value list name:

Keyword (short)	Keyword (long)	FileMaker Pro Equivalent Operator
Ascend	Ascending	Sort a to z, –10 to 10
Descend	Descending	Sort z to a, 10 to –10
Custom		Sort using the value list associated with the field on the layout

Optional with: `–find` or `–findall` requests**Requires:** The `–sortfield` parameter

Example: `FMPPro?–db=employees.fp5&–format= –fmp_xml&–sortfield=First+Name&–sortorder=descend&–find`

–script (Script)**Name/Value Type:** Parameter

What it does: Specifies the FileMaker Pro script that will be performed after finding and sorting records (if specified) during processing of the `–find` request.

Value is: Name of the script to perform.**Optional with:** `–find` or `–findall` requests

Example: `FMPPro?–db=employees.fp5&–format= –fmp_xml&–script=Omit+Script&–find`

–script.prefind (Script before Find)**Name/Value Type:** Parameter

What it does: Specifies the FileMaker Pro script that will be performed before finding and sorting of records (if specified) during processing of the `–find` request.

Value is: Name of the script to perform.**Optional with:** `–find` or `–findall` requests

Example: `FMPPro?–db=employees.fp5&–format= –fmp_xml&–script.prefind=My+Script&–find`

–script.presort (Script before Sort)**Name/Value Type:** Parameter

What it does: Specifies the FileMaker Pro script that will be performed after finding records and before sorting records (if specified) during processing of the `–find` request.

Optional with: `–find` or `–findall` requests

Example: `FMPPro?–db=employees.fp5&–format= –fmp_xml&–script.presort=OmitOne&–find`

–styletype (Style type)

Name/Value Type: Parameter

What it does: Tells the FileMaker Pro Web Companion to generate an XML-stylesheet processing instruction within the grammar—setting the value of the type attribute (type=text/css or type=text/xsl)—so you can use cascading style sheets (CSS) or Extensible Stylesheet Language (XSL) documents with your XML document. This parameter is used in conjunction with the `–stylehref` parameter.

Optional with: All requests

Requires: The `–stylehref` parameter

Example: `FMPPro?–db=employees.fp5&–format= –fmp_xml&–styletype=text/css&–stylehref=mystylesheet.css&–find`

–stylehref (Style href)

Name/Value Type: Parameter

What it does: Tells the FileMaker Pro Web Companion to generate an XML-stylesheet processing instruction within the grammar—setting the value of the href attribute (href=document.css or href=document.xml)—so you can use cascading style sheets (CSS) or Extensible Stylesheet Language (XSL) documents with your XML document. This parameter is used in conjunction with the `–styletype` parameter.

Optional with: All requests

Requires: The `–styletype` parameter

Example: `FMPPro?–db=employees.fp5&–format= –fmp_xml&–styletype=text/xsl&–stylehref=mystylesheet.xml&–find`

–password (Database password)

Name/Value Type: Parameter

What it does: Specifies the database password (set in the Access Privileges > Define Passwords dialog box) to open a database with.

Optional with: The `–dbopen` request

Example: `FMPPro?–db=employees.fp5&–dbopen&–password=dbpassword&–format= –fmp_xml& –styletype=text/css&–stylehref=mystylesheet.css`

field name (Name of specific field)

Name/Value Type: Field name

What it does: Field names are used to control `–find` criteria or to modify the contents of a record. When a value for a specific field needs to be sent to FileMaker Pro, the name portion of the name/value pair is the name of the field in the FileMaker Pro database. Field names used in this manner should not start with the hyphen (`–`) character.

Name is: Name of the field in the database.

Value is: For `–new` and `–edit` requests, the value contains the data for a record. Multiple occurrences of a field allow the data to be put into separate repetitions of a repeating field. For `–find` requests, the value is a find request on the specified field. For all other requests, these name/value pairs are not needed.

Required with: `–new` and `–edit` requests

Optional with: `–find` requests

Example: `FMPPro?–db=employees.fp5&–format= –fmp_xml&–op=eq&FirstName=Sam&–max=1&–find`

Appendix C

FileMaker Pro values for error codes

The FileMaker Pro Web Companion generates an error code for databases published in XML, JDBC, or CDML format every time data is requested. The following table describes the value of each error code, which is based on the type of query to the database. Use these values to do error handling on your web pages.

For more information, see the Status (CurrentError) function described in chapter 11 of the *FileMaker Pro User's Guide* or type `error messages` in the Index tab of FileMaker Pro Help.

Note In FileMaker Pro for Mac OS, if an error occurs while performing an AppleScript from ScriptMaker, the AppleScript error code will be returned.

Error code value	Description
-1	Unknown error
0	No error (success)
1	User canceled action
2	Memory error
3	Command is unavailable (for example, wrong operating system, wrong mode, etc.)
4	Command is unknown
5	Command is invalid (for example, a Set Field script step does not have a calculation specified)
100	File is missing
101	Record is missing
102	Field is missing
103	Relationship is missing
104	Script is missing

Error code value	Description
105	Layout is missing
200	Record access is denied
201	Field cannot be modified
202	Field access is denied
203	No records in file to print or password doesn't allow print access
204	No access to field(s) in sort order
205	Cannot create new records; import will overwrite existing data
206	Cannot change password or file is not modifiable
207	Cannot access field definitions or file is not modifiable
300	The file is locked or in use
301	Record is in use by another user
302	Script definitions are in use by another user
303	Paper size is in use by another user
304	Password definitions are in use by another user
305	Relationship or value list definitions are in use by another user
306	Record modification ID does not match
400	Find criteria is empty
401	No records match the request
402	Selected field is not a match field for a lookup
403	Exceeding maximum record limit for trial version of FileMaker Pro
404	Sort order is invalid

Error code value	Description
405	Number of records specified exceeds number of records that can be omitted
406	Replace/Reserialize criteria is invalid
407	One or both match fields are missing (invalid relationship)
408	Specified field has inappropriate data type for this operation
409	Import order is invalid
410	Export order is invalid
411	Cannot perform delete because related records cannot be deleted
412	Wrong version of FileMaker Pro used to recover file
500	Date value does not meet validation entry options
501	Time value does not meet validation entry options
502	Number value does not meet validation entry options
503	Value in field is not within the range specified in validation entry options
504	Value in field is not unique as required in validation entry options
505	Value in field is not an existing value in the database as required in validation entry options
506	Value in field is not listed on the value list specified in validation entry option
507	Value in field failed calculation test of validation entry option
508	Invalid value entered in Find mode
509	Field requires a valid value
510	Related value is empty or unavailable
511	Value in field exceeds maximum number of allowed characters
600	Print error has occurred

Error code value	Description
601	Combined header and footer exceed one page
602	Body doesn't fit on a page for current column setup
603	Print connection lost
700	File is of the wrong file type for import
701	Data Access Manager can't find database extension file
702	The Data Access Manager was unable to open the session
703	The Data Access Manager was unable to open the session; try later
704	Data Access Manager failed when sending a query
705	Data Access Manager failed when executing a query
706	EPSF file has no preview image
707	Graphic translator cannot be found
708	Can't import the file or need color computer to import file
709	QuickTime movie import failed
710	Unable to update QuickTime file reference because the database is read-only
711	Import translator can not be found
712	XTND version is incompatible
713	Couldn't initialize the XTND system
714	Password privileges do not allow the operation
715	Specified Excel worksheet or named range is missing
716	A SQL query using DELETE, INSERT, or UPDATE is not allowed for ODBC import
717	There is not enough XML/XSL information to proceed with the import or export
718	Error in parsing XML file (from Xalan)
719	Error in transforming XML document (after XSL transformation)

Error code value	Description
720	Error in parsing the XML document (after XSL transformation)
721	Error when exporting--intended format does not support repeating fields
800	Unable to create file on disk
801	Unable to create temporary file on System disk
802	Unable to open file
803	File is single user or host cannot be found
804	File cannot be opened as read-only in its current state
805	File is damaged; use Recover command
806	File cannot be opened with this version of FileMaker Pro
807	File is not a FileMaker Pro file or is severely damaged
808	Cannot open file because access privileges are damaged
809	Disk/volume is full
810	Disk/volume is locked
811	Temporary file cannot be opened as FileMaker Pro file
812	Cannot open the file because it exceeds host capacity
813	Record Synchronization error on network
814	File(s) cannot be opened because maximum number is open
815	Couldn't open lookup file
816	Unable to convert file
817	Unable to open file because it does not belong to this solution
818	FileMaker Pro cannot network for some reason
900	General spelling engine error
901	Main spelling dictionary not installed
902	Could not launch the Help system

Error code value	Description
903	Command cannot be used in a shared file
904	Command can only be used in a file hosted under FileMaker Server
905	No active field selected--command can only be used if there is an active field
950	Adding repeating related fields is not supported
951	An unexpected error occurred
952	Mail format not found (Web Companion)
953	Mail value missing (Web Companion)
971	The user name is invalid
972	The password is invalid
973	The database is invalid
974	Permission Denied
975	The field has restricted access
976	Security is disabled
977	Invalid client IP address
978	The number of allowed guests has been exceeded (for the 10 guest limit over a 12 hour period)

THIS PAGE INTENTIONALLY LEFT BLANK

Index

Numerics

- 3-character filename extensions
 - for runtime databases (Mac OS) 6-8
 - for runtime databases (Windows) 6-8
 - registered by the runtime application 7-5

A

- About <your database> layout
 - adding a custom script 6-2, 6-4, 6-11
 - creating 4-8
 - displaying 4-9
 - example 4-8
 - password warning 4-15
 - prevented modification warning 4-15
 - requirements 4-8
 - using 4-8
- About FileMaker Pro Runtime, default menu command 6-12
- Access Log File option 8-4, 8-14
- access privileges
 - settings for Kiosk mode 4-7
- Access Privileges command
 - denying access to 4-7
 - unavailable in FileMaker Pro 6-2, 6-11
 - unavailable in runtime applications 6-2, A-1
- administrator privileges
 - and Mac OS X installations 3-2
 - and Windows installation 2-1
- amp (&) character entity 5-3
- API (application programming interface)
 - for the FileMaker JDBC Driver 11-8
 - Java, for executing SQL statements 11-1
- apos (') character entity 5-3
- Apple events in runtime applications A-1
- AppleScript
 - error codes C-1

- applets, Java
 - Microsoft XML Data Source Object 10-3
- Application Preferences in runtime applications A-1
- ASCII characters
 - escaping in SQL identifiers 11-7
 - in cross-platform databases 4-12
 - in plug-in IDs 12-8
 - in theme names 5-8
 - in XML documents 10-8
- Asset Management.fp5 database 11-11, 11-15
- attribute values for layout themes 5-7
- automating the Developer Tool 6-15
- auxiliary files
 - opening 4-2
 - updating 7-7

B

- Balloon Help (Mac OS) 4-8
- basic Java classes 11-1, 11-8
- Binder utility. *See* FileMaker Developer Tool
- binding
 - databases to runtime applications 6-2, 6-6, 7-1
 - files for cross-platform runtime solutions 6-9
- binding key for runtime database solutions 6-7
- BMP graphic import filter 7-2
- body layout parts 5-4
- Break encoding parameter 9-12
- Browse mode in runtime applications A-1
- built-in home page, displaying on the Web 8-5
- buttons
 - creating 4-6
 - for closing Help windows 4-9
 - showing attached scripts 4-6

C

- C/C++ 12-1
- cache size
 - in Mac OS X 3-3
 - in Windows 2-4
- calculations, using external functions 8-15
- CALL stored procedure SQL statements 11-4
- cascading style sheets (CSS) 10-10, 10-11, 10-13
- CDML
 - action tags 9-4, 9-8
 - CGI requests 9-3
 - described 8-2, 9-1
 - encoding parameters 9-12
 - error pages 9-11
 - examples 9-1, 9-14, 9-15
 - FMP-Include tag 9-16
 - format files 9-1, 9-2
 - replacement tags 9-8
 - templates 9-6
 - token tags 9-16
 - variable tags 9-4, 9-8
- CDML Reference database
 - described 9-1, 9-9
 - using 9-9
- CDML Tool
 - choosing encoding parameters 9-6
 - described 9-1, 9-5
 - displaying tag syntax 9-6
 - parameters and value list options 9-6
 - tag categories 9-8
 - Tags tab 9-6
 - Templates tab 9-5
 - using 9-5
- centering database windows in Kiosk mode 4-6
- CGI requests.
 - See* FileMaker Pro CGI requests

- CGM graphic import filter 7-2
 - character entities, XML 5-3
 - character escaping 11-4
 - characters, platform-specific 4-12
 - Claddon.clr spelling dictionary 7-3
 - COLOR attribute 5-7
 - color palette
 - FileMaker Pro 5-10
 - for cross-platform databases 4-13
 - color values for layout themes 5-10
 - com.fmi.jdbc.JdbcDriver driver class 11-2
 - command line, installing from 2-2
 - comment tags
 - in CDML templates 9-7
 - in Java examples 11-11
 - in layout themes 5-10
 - Common Gateway Interface (CGI)
 - See also* FileMaker Pro CGI requests described 8-2
 - Common Log Format 8-14
 - compression utilities for runtime databases 7-4
 - configuring external function plug-ins 12-3, 12-9, 12-12
 - container fields 8-16
 - converting files to FileMaker Pro 6 1-4
 - Creating Dynamic Buttons example 4-6
 - creator codes 6-8
 - cross-platform databases
 - binding 6-9
 - character sets 4-12
 - color palettes 4-13
 - design tips 4-11
 - fonts 4-11
 - graphics 4-13
 - outline and shadow text styles 4-11
 - platform-specific scripts 4-15
 - QuickTime movies 4-13
 - scripts for printing 4-14
 - showing and hiding status bar (Windows) 4-14
 - Status (CurrentPlatform) function 4-14
 - testing 4-16
 - text layouts 4-12
 - CSS and XML 10-11
 - custom
 - About command 6-11, A-1
 - About layouts 4-8, 4-9
 - databases, testing 4-16
 - Help command 6-13, A-1
 - Help layouts 4-8, 4-9
 - installers for runtime databases 7-4
 - layout themes 5-1
 - Scripts menu name 6-14
 - custom error pages using CDML 9-11
 - custom home page
 - creating with JavaScript 8-7
 - described 9-2
 - displaying on the Web 8-5
 - custom web portal example 8-7
 - custom web publishing
 - using a database layout 8-1
 - using CDML 8-1, 8-2, 9-2
 - using Java 11-1
 - using XML 8-1, 10-1
- D**
- damaged runtime files, recovering 7-6
 - data import files
 - for runtime databases (Mac OS) 7-3
 - for runtime databases (Windows) 7-2
 - data type mapping 11-4, 11-7
 - Database Design Report 1-1, 4-1
 - XML grammar 4-1
 - database error codes 10-3
 - databases
 - cross-platform 4-11
 - opening runtime 7-5
 - preparing 4-2, 4-11
 - preventing modification 4-7
 - recovering damaged 7-6
 - renaming 6-10
 - db request parameter B-6
 - dBASE import filter 7-2, 7-3
 - dbnames requests B-3
 - DbOpen and DbClose pseudo procedures 11-5
 - dbopen and -dbclose requests 8-18, B-5
 - debugging
 - external function plug-ins 12-13
 - decoding and repairing passwords 4-15
 - default home page, setting 8-6
 - default values for layout themes 5-9
 - default XML namespace declarations 10-2
 - default.htm 8-5
 - Define Fields command
 - unavailable in FileMaker Pro 6-2, 6-11
 - unavailable in runtime applications 6-2, A-1
 - Define Relationships command
 - unavailable in FileMaker Pro 6-2, 6-11
 - unavailable in runtime applications 6-2, A-1
 - Define Value Lists command
 - unavailable in runtime applications 6-2, 6-9, A-1
 - delete requests B-3
 - DELETE SQL statements 11-4, 11-6
 - design tips
 - for cross-platform files 4-11
 - desktop conflicts with filename extensions 6-8
 - Developer Tool. *See* FileMaker Developer Tool
 - developer-tool-independent example 11-8
 - dictionaries 1-5
 - dictionaries.
 - See* spelling dictionaries
 - dictionary
 - in runtime database solution 4-3
 - differences between runtime applications and FileMaker Pro A-1
 - Display encoding parameter 9-12
 - distributing runtime databases 7-1, 7-4

- Do_Init function 12-11
- Document Object Model (DOM)
 - Microsoft 10-3
 - W3C 10-16, 10-17
- Document Preferences in runtime applications A-2
- document type definitions (DTDs) 10-2, 10-3, 10-5
- documenting
 - custom databases 4-8, 4-10
 - external function plug-ins 12-13
 - runtime databases 7-5, 7-6
- double colons in portal fields
 - in CGI requests 9-4, 10-9
 - in XML documents 10-4
- driver properties, for FileMaker JDBC URL 11-3
- DRW graphic import filter 7-2
- dso_xml format parameter 10-3
- Dynamic Link Libraries (DLLs) 7-1, 7-2

E

- edit requests
 - CDML examples 9-4
 - XML examples 10-10, B-3
- editing multiple records in portals 9-4, 10-10
- EFFECT attribute 5-7
- elements
 - in FMPDSORESULT grammars 10-4
 - in FMPXMLLAYOUT grammars 10-7
 - in FMPXMLRESULT grammars 10-5
 - in layout themes 5-3
- Employee Database CDML example 9-13, 9-14
- emulating interface elements 4-6
- encoded XML 10-3, 10-8
- encoding parameters for CDML 9-6, 9-12
- encrypting data 11-3
- Error Log File option 8-4, 8-14

- errors
 - creating error messages 9-11
 - during binding 6-6
 - errnum CDML variable tag 9-11
 - error CDML variable tag 9-11
 - Error folder 9-11
 - error pages recognized by the Web Companion 9-11
 - specifying FileMaker Pro error codes 9-11, 10-3, C-1
- escape driver property 11-4
- escaping of lower ASCII characters 11-7
- example CGI requests for XML B-1
- example Plug-In file 12-4
- examples
 - comparing CSS, XSL, and JavaScript 10-11
 - Creating Dynamic Buttons 4-6
 - custom About layouts 4-8
 - custom Help layouts 4-9
 - Employee Database 9-14
 - FileMaker Pro Explorer 11-8
 - generated FMPDSORESULT grammar 10-4
 - generated FMPXMLLAYOUT grammar 10-7
 - generated FMPXMLRESULT grammar 10-6
 - Guest Book 9-14
 - JBuilder Inventory 11-11
 - Kiosk Solution Example 4-4, 6-1
 - registering the FileMaker JDBC Driver 11-3
 - Relational Example 4-1
 - Runtime Solution Example 6-1
 - Shopping Cart 9-15
 - Visual Cafe Inventory 11-15
 - XML and CSS 10-13
 - XML and JavaScript 10-16
 - XML and XSLT 10-14
 - XML Inventory 10-17
- Excel import filter 7-3

- exporting data into HTML tables 8-16
- Extensible Markup Language (XML). *See* XML
- Extensible Stylesheet Language (XSL). *See* XSL stylesheets
- Extensible Stylesheet Language–Transformations (XSLT). *See* XSLT example
- extensions, filename 6-8
- External Function message 12-12
- external function plug-ins
 - configuring 12-3, 12-9, 12-12
 - debugging 12-13
 - documenting 12-13
 - enabling 12-3
 - function name prefix 12-10
 - in runtime applications A-1
 - main entry point 12-9
 - messages sent by FileMaker Pro 12-10
 - plug-in ID 12-9
 - registering with FileMaker, Inc. 12-1, 12-13
 - required feature string syntax 12-8
 - resource ID conflicts (Mac OS) 12-13
 - Specify Calculations dialog box 12-9
 - Web Companion 8-15
- external functions
 - for web site monitoring 8-15
 - in calculation fields and scripts 8-15, 12-1

F

- feature string syntax for external function plug-ins 12-8
- fetchsize driver property 11-4
- field name request parameter B-10
- field names, spaces in 10-4, 11-3, 11-13
- file naming requirements 6-10
- file sharing with runtime applications 7-5
- FileMaker Developer
 - customer support 1-1
 - documentation 1-5

- hardware and software requirements 1-2
- installation code 1-2
- installing (Mac OS) 3-1
- installing (Windows) 2-1
- license agreement 1-6
- registering 1-1
- FileMaker Developer Tool
 - automating the process 6-15
 - described 6-1
 - generated DLLs (Windows) 7-2
 - generated runtime files 7-1
 - options, described 6-2
 - saving option settings 6-14
 - specifying files 6-3
 - using 6-2
- FileMaker JDBC Driver
 - data type mapping 11-7
 - described 11-1
 - driver class and main entry point 11-2
 - escaping lower ASCII characters 11-7
 - examples 11-8, 11-11, 11-15
 - FileMaker extensions 11-8
 - implementing JDBC 1.2 API 11-2
 - including with FileMaker Pro Java applications and applets 11-2
 - JDBC interfaces 11-8
 - opening and closing databases 11-5
 - registering with the JDBC driver manager 11-2, 11-3
 - specifying the JDBC URL 11-2
 - SQL support 11-4
 - using 11-2
 - working with JDK 1.1 and Java 2 11-2
- FileMaker Pro 4.0 Developer Edition, upgrading from 4-16
- FileMaker Pro CGI requests
 - adding records to portals 9-4, 10-9
 - and script buttons 8-9
 - CDML request and parameter names 9-4 described 8-2
 - editing multiple records in portals 9-4, 10-10
 - example CGI requests for XML B-1
 - for accessing databases 8-2
 - for CDML 9-3
 - for XML 10-2, 10-8
 - using JavaScript 8-8, 10-16
 - XML request and parameter names 10-8
- FileMaker Pro Explorer application
 - described 11-8
 - starting 11-9
- FileMaker Pro Help
 - See also* Help
 - described 4-8
- FileMaker Pro plug-ins. *See* external function plug-ins
- FileMaker Pro themes. *See* layout themes
- FileMaker Server 7-5
- FileMaker, Inc.
 - registering external function plug-ins 12-1, 12-13
- FileMakerExplorer.java file 11-10
- filename extensions
 - adding or removing .fp5 6-14
 - desktop conflicts (Mac OS) 6-8
 - for runtime databases 6-8
 - registered by runtime applications 7-5
 - registry conflicts (Windows) 6-8
- filenames for instant web pages 8-13
- files
 - closing in Kiosk mode 4-4
 - converting to FileMaker Pro 6 1-4
 - cross-platform 4-11
 - opening runtime 7-5
 - preparing 4-2, 4-11
 - recovering damaged 7-6
 - renaming 6-10
- Find mode in runtime applications A-1
- find requests 10-13, 10-14, B-1
- FMFlags.h header file 12-7
- FMP.js JavaScript library 10-16, 10-17, 10-21
- FMPDRESULT grammars 10-3
- FMP-Include tag 9-16
- Fmpjdbc12.jar file 11-2
- FMP-Log tag 8-15
- FMPRT.hlp file (Windows) 7-1
- FMPXMLLAYOUT grammars 10-7
- FMPXMLRESULT grammars 10-6
- FMTHEME element 5-2
- FMX filename extension 12-3
- FONT attribute 5-8
- Fonts folder, including with runtime databases (Mac OS) 7-3
- fonts for cross-platform databases 4-11
- footer layout parts 5-5
- Format encoding parameter 9-12
- format files
 - and web styles 8-13
 - described 9-2
 - templates, using 9-6
- format request parameter B-7
- fp5 filename extension, adding 6-14
- frame types 2-3
- function name prefix for external function plug-ins 12-10
- functions
 - See also* external functions
 - Status (CurrentError) C-1
 - Status (CurrentPlatform) 4-14
 - Web- external functions 8-16
 - Xpl- external functions 12-4, 12-5

G

- Go to Layout script step 4-6
- grammars. *See* XML
- graphic import files
 - for runtime databases (Mac OS) 7-3
 - for runtime databases (Windows) 7-2
- graphics in cross-platform databases 4-13
- gt (greater than) character entity 5-3
- Guest Book CDML example 9-13, 9-14

H

Handles 12-12

header layout parts 5-4

Help

- Balloon Help (Mac OS) 4-8
- button on toolbar in runtime applications A-3
- custom layout 4-8, 4-9
- custom menu command 6-2, 6-13, A-1
- documentation for custom databases 4-8, 4-10
- examples of custom layouts 4-9
- FileMaker Pro Help
- Status Bar Help (Windows) 4-8
- What's This? Help (Windows) 4-10

Help layout

- adding a custom script 6-2, 6-13

hexadecimal (hex) values for layout themes 5-10

HINT attribute 5-2, 5-8

home button in browser 8-13

home pages

- built-in 8-5
- custom 8-1, 8-5

hosting multiple web sites 8-6

HREF links 8-2, 9-3, 10-2, B-1

HTML

- described 10-1
- encoding parameter for CDML format files 9-12
- form elements 9-8
- form examples for XML B-1
- format 10-2
- forms 8-2, 9-4, 10-2
- hidden INPUT tags 9-8
- tables, exporting data into 8-16
- translating information into 8-15
- with XML and JavaScript 10-16

HTTP (Hypertext Transfer Protocol)

- translating information into 8-15
- using with FileMaker JDBC Driver 11-2

I

Idle message 12-11

If script step 4-14

Import script step 4-17

importing records 4-17

index.htm 8-5

Information Log File option 8-4, 8-15

Initialization message 12-11

INSERT SQL statements 11-4

installation log file 2-2, 3-2

installers

- for custom database solutions 4-2
- for runtime databases 7-4

installing

- FileMaker Developer (Mac OS) 3-1
- FileMaker Developer (Windows) 2-1
- runtime database solutions 7-5

InstallShield 7-4

Instant Web Publishing

- custom home pages with 8-1
- described 8-2
- script steps supported 8-9

interface elements, emulating 4-6

internal reference links, updating 6-10

Internet Explorer 4.0 10-3

Internet Explorer 4.5 8-13

Internet Explorer 5.0 10-12, 10-17

Inventory.fp5 database 10-17, 11-11

inventory_db database 11-15, 11-17

IP addresses 8-4, 9-2

- displaying with CDML 9-9
- returned by external function 8-16

IP Guest Limit 8-2

IPX/SPX 2-3

J

Java 2 environments 11-2

Java applets

- Microsoft XML DSO 10-3

Java applications

- FileMaker Pro database-aware 11-1
- FileMaker Pro Explorer example 11-8
- JBuilder Inventory example 11-11
- Visual Cafe Inventory example 11-15

Java classes

- basic 11-1, 11-8

Java Development Kit (JDK) 11-2, 11-8

JavaScript

- and custom home page 8-7
- and FileMaker Extended XML 10-2
- example 10-16, 10-17
- versus CSS or XSL 10-11
- with Internet Explorer 4.0 10-3

JBuilder 3.0 Professional for Windows 11-1, 11-11

Jbuilder.ini file 11-11

JDBC

- See also* FileMaker JDBC Driver
- described 11-1

JDBC 1.2 API 11-2

JDBC URL 11-2

K

key, binding 6-7

Kiosk mode

- access privileges for opening 4-7
- centering database windows 4-6
- closing files 4-4
- Developer Tool option 6-2
- displaying runtime databases 4-3, 6-9
- preparing databases 4-4

Kiosk Solution Example 4-4, 6-1

L

-lay request parameter B-6

Layout Mode command

- unavailable in FileMaker Pro 6-2, 6-11
- unavailable in runtime applications 6-2, A-1

- layout parts, XML elements for 5-4
- layout themes
 - attribute values 5-7
 - attributes for single-line elements 5-6
 - basic requirements 5-3
 - checking for errors 5-11
 - color values 5-10
 - comment tags 5-10
 - creating 5-1
 - default values 5-9
 - elements for layout parts 5-4
 - elements for layout text 5-5
 - modifying 5-1
 - multi-line elements 5-4, 5-5
 - pattern values 5-9
 - single-line elements 5-4, 5-5
- layoutnames requests B-4
- layouts
 - About layout 4-8, 6-12
 - cross-platform 4-11
 - Help layout 4-9
- leading grand summary layout parts 5-4
- leading subsummary layout parts 5-4
- license agreement
 - for FileMaker Developer 1-6
 - for ODBC driver 1-6
- links
 - HREF 8-2, 8-6, 9-3, 10-2
 - testing 8-18
 - to CDML format files 9-2
 - to IP addresses 8-6
 - XML example requests B-1
- Local Data Access Companion plug-in A-1
- localhost networking 8-18
- log files
 - generated by the Web Companion 8-14
 - installation 2-2, 3-2
- lop request parameter B-7
- Lotus SLD graphic import filter 7-2
- lt (less than) character entity 5-3

M

- Mac OS X
 - enabling Web Companion 3-5
 - FileMaker Pro features not supported 1-4
 - Help tags 4-8
- MacPaint graphic import filter 7-2
- main entry point for external function plug-ins 12-9
- mapping
 - colors for cross-platform databases 4-13
- mapping data type 11-4, 11-7
- master passwords 4-7
- max request parameter B-8
- memory 2-4, 3-3
- menu commands
 - available in FileMaker Pro A-3
 - available in runtime applications A-3
 - custom About layout 6-2, 6-4, 6-11, A-1
 - custom Help layout 6-2, 6-4, 6-13, A-1
 - default About runtime layout 6-12
 - permanently unavailable in FileMaker Pro 6-2, 6-11
 - unavailable in runtime applications 6-2, 6-9, A-1
- messages
 - error, examples of 9-12
 - errors sent by the Web Companion 9-11, C-1
 - sent to external function plug-ins 12-10
- Metafile graphic import filter 7-2
- Microsoft Notepad 10-20
- Microsoft XML Data Source Object (DSO) 10-2, 10-3
- MIME (Multipurpose Internet Mail Extensions) types 8-13
- MindVision Installer VISE 7-4
- minimum requirements for runtime database solutions 7-4
- ModId pseudo column 11-4
- modid request parameter B-7

- modifying FileMaker Pro layout themes 5-1
- monitoring web sites
 - using external functions 8-15
 - using log files 8-14
- MRJ 2.2 11-9
- multi-line elements for layout themes 5-4, 5-5

N

- namespaces, XML
 - described 10-3
 - for the FMPDSORESLT grammar 10-4
 - for the FMPXMLLAYOUT grammar 10-7
 - for the FMPXMLRESULT grammar 10-6
- naming runtime database solutions 6-7
- navigating in Kiosk mode 4-4
- NCSA/CERN-compatible Common Log Format 8-14
- networking
 - requirements 1-3
- networks
 - protocol 2-3, 3-3
- New Database script step A-6
- New Database tool A-3
- New Layout/Report assistant 5-1, 5-2, 5-5
- new requests
 - CDML examples 9-4
 - XML examples 10-9, B-2
- Note 9-4
- Notepad 11-10, 11-11

O

- ODBC (Open Database Connectivity) driver
 - See also* JDBC, FileMaker JDBC Driver
 - installing in Mac OS X 3-2
 - Java equivalent 11-1
- OLE automation in runtime applications A-1
- onClick JavaScript 8-8
- onClick scripting event 10-16
- onLoad event handler 10-16

ONOFF attribute 5-8
 onscreen Help. *See* Help
 -op request parameter B-8
 Open Define Fields script step A-6
 Open Define Relationships script step A-6
 Open Define Value List script step A-6
 Open File script step 4-17, A-6
 Open Help script step A-6
 Open ScriptMaker script step A-6
 Open Sharing script step A-6
 Open tool A-3
 opening

- files in runtime applications 4-2, 7-5
- runtime databases in FileMaker Pro 4-3, 4-7, 6-9

 outline and shadow text styles 4-11

P

Page Setup/Print Setup script steps 4-14
 PARTNUMBER attribute 5-8
 password driver property 11-4
 passwords

- decoding and repairing 4-15
- default 4-7
- master 4-7
- protecting files from hackers 6-11
- required warning in About layouts 4-15

 PATTERN attribute 5-8
 pattern palette, FileMaker Pro 5-9
 pattern values for layout themes 5-9
 PCX graphic import filter 7-2
 People.fp5 database 10-12
 people_form.css stylesheet 10-13
 people_form.htm page 10-16
 people_form.xsl stylesheet 10-15
 Perform External Script script step A-6
 permanently preventing modification of databases

- option in the Developer Tool 4-7, 6-2
- required warning in About layouts 4-15
- unavailable menu commands 6-11

 PIC graphic import filter 7-2
 platform-specific scripts 4-15
 plug-ins

- See also* external function plug-ins
- IDs 12-8, 12-9
- in runtime applications A-1
- Local Data Access Companion A-1
- registering 12-13
- registering ID 12-8
- Remote Data Access Companion A-1
- Web Companion 8-2, A-1

 port

- setting for Web Companion 3-5

 portals

- adding records 9-4, 10-9
- editing multiple records 9-4, 10-10

 Preferences message 12-12
 preferences, in runtime applications A-1
 preparing databases

- for cross-platform use 4-11
- for Kiosk mode 4-4
- for runtime database solutions 4-2

 Preview mode in runtime applications A-1
 primary file

- connecting auxiliary files 4-1
- determining custom solution name 6-5
- displaying About layout 4-9
- in Kiosk mode 4-4
- in Relational Solution Example 4-1
- in runtime database solutions 6-6
- problems with double-clicking 7-5
- specifying for custom database solutions 6-4
- stored setting for Scripts menu name 6-14
- updating 7-7

 primary key. *See* RecordId pseudo column
 Print Field Definitions script step A-6
 Print Script Definitions script step A-6

Print script step 4-14
 printing

- from runtime applications 4-3
- script steps for 4-14
- setting document margins 4-3

 protecting databases

- by removing command access 6-2
- in subfolders of the Web folder 8-18
- with an About layout 4-8
- with record-level passwords 8-4
- with remote administration passwords 8-18
- with Web Security Database 8-4

 publishing on the Web

- See also* Web Companion
- comparing static HTML form with dynamic CDML 9-14
- database error codes C-1
- described 8-1
- encrypting data 11-3
- error pages 9-11
- exporting data into HTML tables 8-1, 8-16
- FileMaker Pro database-aware Java applets 8-1, 11-1
- FileMaker Pro databases 8-1
- read-only databases 10-2
- requirements 1-3
- using CDML 8-1, 8-2, 9-2
- using custom home pages 8-5
- using Instant Web Publishing 8-2
- using JDBC 8-1
- using XML 8-1, 10-2

Q

QuickTime

- movies in cross-platform databases 4-13

 quot (") character entity 5-3

- R**
- Rapid Application Development (RAD) tools
 - 8-1, 11-1, 11-2
 - Raw encoding parameter 9-12
 - recid request parameter B-7
 - RecordId pseudo column 11-4
 - recovering damaged runtime files 7-6
 - registering
 - FileMaker Developer 2-2
 - FileMaker JDBC Driver 11-2
 - FileMaker Pro external function plug-ins
 - 12-1, 12-13
 - plug-in ID 12-8
 - registry
 - conflicts with filename extensions 6-8
 - settings stored for runtime applications A-6
 - Relational Example 4-1
 - Remote Administration option 8-4
 - Remote Data Access Companion plug-in A-1
 - renaming
 - database files 6-10
 - Scripts menu 6-14, A-1
 - Solution Options file 6-15
 - requesting data. *See* FileMaker Pro CGI requests
 - requirements
 - for About layouts 4-15
 - for About <your database>solution 4-8
 - resource ID conflicts (Mac OS) 12-13
 - restoring
 - document and application icons (Mac OS)
 - 6-8
 - document icons (Windows) 6-8
 - Restrict access to IP addresses option 8-4
 - restricting access to menu commands 6-11
 - revising registered external function plug-ins 12-14
 - runtime applications
 - See also* runtime databases, runtime database solutions
 - and Web Companion A-1
 - Apple events in A-1
 - Application Preferences A-1
 - compared to FileMaker Pro A-1
 - Document Preferences A-2
 - FileMaker Pro file sharing 7-5
 - ignored script steps A-6
 - registering filename extensions 7-5
 - starting 7-5
 - stored registry settings A-6
 - toolbars A-3
 - unavailable menu commands 6-2, 6-9, A-1
 - What's This? Help (Windows) 7-1
 - runtime database solution
 - spell checking 4-3
 - runtime database solutions
 - See also* runtime applications, runtime databases
 - About layout example 4-8
 - About layout requirements 4-8, 4-15
 - compression utility applications 7-4
 - creating 6-6
 - custom About layout 6-12
 - custom installers 7-4
 - distributing updates 7-4, 7-6
 - distribution requirements 4-15
 - documenting 7-5, 7-6
 - generated files by the Developer Tool 7-1
 - methods of distribution 7-1
 - naming 6-8
 - preparing files 4-2
 - recovering damaged files 7-6
 - required DLL files (Windows) 7-2
 - supplemental runtime files 7-1, 7-2, 7-3
 - runtime databases
 - automating the binding process 6-15
 - displaying in Kiosk mode 4-3
 - importing records 4-17
 - opening in FileMaker Pro 4-7, 6-9
 - opening in runtime applications 4-2
 - printing reports 4-3
 - upgrading 4-16
 - Runtime Files folder, contents of 7-1
 - Runtime Solution Example
 - About layout example 4-8
 - described 6-1
- S**
- schemas. *See* XML grammars
 - screenshot 6-4, 6-5
 - Script Debugger 1-1, 4-1
 - script request parameter B-9
 - script steps ignored by runtime applications
 - A-6
 - script.prefind request parameter B-9
 - script.presort request parameter B-9
 - scripting languages
 - See also* JavaScript
 - JavaScript 10-3
 - VBScript 10-3
 - ScriptMaker command
 - unavailable in FileMaker Pro 6-2, 6-11
 - unavailable in runtime applications 6-2, A-1
 - scriptnames requests B-4
 - scripts
 - attaching to About and Help commands 4-8
 - attaching to buttons 4-6
 - for cross-platform printing 4-14
 - for emulating menu commands and window controls 4-6
 - for FileMaker Pro CGI requests 10-2
 - for importing 4-17
 - for navigating in Kiosk mode 4-4
 - for opening runtime databases 4-2, 4-17
 - in layout for custom home page 8-8
 - platform-specific 4-15
 - startup 4-5, 4-6
 - using external functions 8-15
 - using the Solution Options file 6-15
 - Scripts menu
 - accelerator key for new name 6-14
 - renaming 6-14, A-1

- searching
 - databases on the Web 8-1, 9-14
 - TechInfo database 1-6
- security 8-3, 8-4
- SELECT SQL statements 11-4, 11-6
- Set Multi-User script step A-6
- Set Zoom Level script step 4-6
- setting document margins for printing 4-3
- setting up single-machine networks 8-18
- shadow and outline text styles 4-11
- sharing
 - databases via the Web Companion 8-5
 - runtime files over a network 7-5
- Shopping Cart CDML example 9-13, 9-15
- Shutdown message 12-11
- SIDES attribute 5-8
- SimpleText 11-10
- single-line elements for layout themes 5-4, 5-5
- SIZE attribute 5-8
- skip request parameter B-8
- Solution Extras folder for runtime databases (Mac OS) 7-3
- Solution Options file
 - modifying for auto-customizing 6-15
 - renaming the file 6-15
 - using 6-15
- Solutions Development Kit (SDK) for FileMaker Pro 3.0 4-16
- solutions. *See* runtime database solutions
- sortfield request parameter B-9
- sortorder request parameter B-9
- spaces in field names 10-4, 11-3, 11-13
- specifying files for the Developer Tool 6-3
- spell checking
 - in runtime solution 4-3
- spelling checker files, for runtime databases (Windows) 7-2
- spelling dictionaries
 - for runtime databases (Mac OS) 7-3
 - for runtime databases (Windows) 7-2

- SQL statements
 - delivered by JDBC drivers 11-1
 - supported by FileMaker JDBC Driver 11-4
- SQL-92 Entry Level compliant 11-2
- startup scripts
 - creating 4-5
 - to display custom layout 6-6
- static web publishing 8-1, 8-16, 9-14
- Status (CurrentError) function C-1
- Status (CurrentPlatform) function 4-14
- Status Bar Help (Windows) 4-8
- status bar, showing and hiding 4-14
- StuffIt 7-4
- STYLE attribute 5-9
- stylehref request parameter B-10
- stylesheets
 - CSS 10-10, 10-11, 10-13
 - described 10-2
 - XML-stylesheet processing instruction 10-2, 10-10
 - XSL 10-10, 10-11
 - XSLT 10-14
- styletype and -stylehref parameters 10-10
- styletype request parameter B-10
- subtype, for FileMaker JDBC URL 11-2
- subprotocol, for FileMaker JDBC URL 11-2
- suppressing Instant Web Publishing controls 8-11
- Swing 1.1.1 11-1, 11-8
- Symantec Visual Cafe 11-1
- System folder for runtime databases (Windows) 7-2

T

- TCP/IP 1-3
- TCP/IP port number 8-4
 - Mac OS X 3-5
- TechInfo database
 - described 1-6

- technical support
 - for FileMaker Developer 1-1
 - for runtime applications 1-6
- templates
 - CDML 9-6
- testing
 - cross-platform databases 4-16
 - custom database solutions 4-16
- text editors 5-1
- text styles, outline and shadow 4-11
- THEMEDEFAULT element 5-9
- THEMENAME element 5-2
- themes. *See* layout themes
- three-character filename extensions
 - for runtime databases 6-8
 - registered by runtime applications (Windows) 7-5
- TIFF graphic import filter 7-2
- title footer layout parts 5-5
- title header layout parts 5-4
- Toggle Status Area script step 4-6, 8-12
- Toggle Window script step 4-6
- token tags 9-16
- toolbars in runtime applications A-3
- trailing grand summary layout parts 5-5
- trailing subsummary layout parts 5-4
- trapping errors C-1

U

- Ukenglsh.mpr spelling dictionary 7-3, 7-4
- Unicode characters 10-8, 11-7
- uninstalling FileMaker Developer (Windows) 2-2
- UPDATE SQL statements 11-4, 11-6
- updating
 - internal reference links 6-10
 - runtime database solutions 7-6
- upgrading runtime databases 4-16
- URL encoding parameter 9-12

URLs (Uniform Resource Locators)
 accessing Java resources 11-2
 FileMaker JDBC URL 11-2
 for FileMaker Pro CGI requests 10-2
 using to access databases 8-1
Usenglish.mpr spelling dictionary 7-3, 7-4
user driver property 11-4
User.upr spelling dictionary 7-3, 7-4
UTF-8 (Unicode Transformation 8 Bit) format 10-8

V

values for error codes C-1
VBScript 10-3
-view requests B-2
Visual Cafe 4.0 Expert Edition for Windows 11-1, 11-15

W

W3C Document Object Model 10-16, 10-17
web browsers
 for XML Inventory example 10-17
 for XML Simple Examples 10-12
 receiving static files 8-2
Web Companion
 See also publishing on the Web
 and runtime applications A-1
 built-in home page 8-5
 configuration options 8-3
 default home page, setting 8-6
 described 8-2
 enabling 8-3
 enabling in Mac OS X 3-5
 error messages automatically generated 9-11
 error pages recognized 9-11
 external functions 8-15
 generated error codes C-1
 generated log files 8-14
 generating XML documents 10-1, 10-2

 generating XML DTDs 10-3, 10-5
 IP Guest Limit 8-2
 monitoring web sites 8-14
 remote administration 8-4
 restricting access to IP addresses 8-4
 sharing databases 8-5
Web- external functions 8-16
web portal example 8-7
web security 8-3, 8-4
Web Security.pdf 1-5
web sites
 Apple 4-13, 6-8
 Apple Developer Support 12-8
 Borland/Inprise (Corel) 11-11
 FileMaker support pages 8-1
 hosting multiple 8-6
 Javasoft 11-7
 monitoring 8-14, 8-15
 Sun Microsystems 11-8
 testing without a network connection 8-17
 W3C organization 8-14
web styles 8-11
WebPortal object 8-7
web-safe colors 5-7, 5-10
What's This? Help (Windows) 4-10, 7-1
WinZip for Windows 7-4
WML (Wireless Markup Language)
 documents 8-13
World Wide Web Consortium (W3C) 8-14

X, Y, Z

XML

 CGI requests 10-2, 10-8
 character entities 5-3
 described 10-1
 document type definitions (DTDs) 10-2, 10-3, 10-5
 documents for layout themes 5-1
 documents generated by the Web Companion 8-1

 -edit request example 10-10
 editors 5-1
 encoded using UTF-8 format 10-3, 10-8
 examples comparing CSS, XSL, and JavaScript 10-11
 FMPDSORESULT grammars 10-3, 10-4
 FMPXMLLAYOUT grammars 10-7
 FMPXMLRESULT grammars 10-5
 format parameters 10-2, 10-3
 grammar for Database Design Report 4-1
 grammars (or schemas), described 10-2
 import filter, Mac OS 7-3
 import filter, Windows 7-3
 namespaces 10-3, 10-4, 10-6, 10-7
 -new request example 10-9
 parsers 10-2, 10-8
 using CSS 10-10
 using XSL 10-10
 XML 1.0 specification 5-3, 10-2
 XML Inventory example 10-17
 XML-document processing instruction 5-3
 XML-stylesheet processing instruction 10-2, 10-10
 XSLT example
 XPAL files for cross-platform use 7-2
 Xpl- external functions 12-4
 XSL stylesheets 10-10, 10-11
 XSLT example 10-14