FileMaker Developer 7

Developer's Guide

FileMaker, Inc.
5201 Patrick Henry Drive
Santa Clara, California 95054

All persons and companies listed in the examples are purely fictitious and any resemblance to existing persons and companies is purely coincidental.

Credits are listed in the Acknowledgements document provided with the software.

For more information, visit our web site at www.filemaker.com.

Edition: 01A

Contents

Chapter 5
Customizing database solutions

Chapter 6
Creating custom layout themes

Chapter 7
Developing third-party FileMaker plug-ins

Chapter 1
Getting started

Welcome to FileMaker® Developer. FileMaker Developer is the Developer edition of FileMaker Pro. It includes advanced features for developers. You can use either FileMaker Pro or FileMaker Developer to create and test your database solutions. You use FileMaker Developer to transform your database solution into a runtime or kiosk-mode application that you can distribute to your users.

About FileMaker Developer

FileMaker Developer includes a number of productivity features designed especially for database developers. They are the Script Debugger, the Database Design Report feature, the File Maintenance feature, and the Custom Functions feature. These features are documented in "Using FileMaker Developer features" on page 11.

You can produce databases for distribution without leaving FileMaker Developer. The Developer Utilities are an integrated component of FileMaker Developer. Not only do the Utilities enable you to produce runtime databases for distribution, they also allow you to rename files while still maintaining links, to prevent users from modifying the design of your databases, and to customize your solutions. The Developer Utilities are documented in "Using the Developer Utilities" on page 17.

Using the FileMaker Developer documentation

This *Developer's Guide* is one component in a comprehensive documentation suite provided with FileMaker Developer. Some of the documents are provided in print and in portable document format (PDF), while others are available in PDF only. FileMaker Developer also comes with a complete online Help system.

This guide assumes that you are familiar with FileMaker Pro or FileMaker Developer, and that you have created a database solution that you want to work on using the FileMaker Developer features.

Note Throughout the documentation, 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.

The following manuals are included:

- FileMaker Developer *Developer's Guide* (this manual): describes how to use the features available in FileMaker Developer
- *Installation and New Features Guide for FileMaker Pro and FileMaker Developer*: contains installation instructions and new features information

- *FileMaker Pro User's Guide*: contains key concepts and basic procedures
- *FileMaker Pro Tutorial*: contains lessons that teach you how to use the major features of FileMaker Pro
- *FileMaker Instant Web Publishing Guide*: describes how to make FileMaker Pro and FileMaker Developer databases accessible to web browser users over an intranet or the Internet
- *Converting FileMaker Databases from Previous Versions*: contains information about converting databases from previous versions to FileMaker Pro 7 and FileMaker Developer 7 format
- *Customizing the FileMaker Pro Templates*: describes how to customize the included database template files for your own needs
- *FileMaker Pro Security Guide*: describes security concerns to keep in mind when using FileMaker products
- *Installing FileMaker ODBC and JDBC Client Drivers*: describes how to install driver files for using FileMaker products with OBDC and JDBC applications
- *FileMaker ODBC and JDBC Developer's Guide*: describes how to use ODBC and JDBC with FileMaker products

Where to find PDF documentation

Most PDF manuals are located in the folder where you installed FileMaker Developer. If you installed FileMaker Developer in the default folder location, the PDF manuals are located here:

- **Windows**: C:\Program files\FileMaker\FileMaker Developer 7\English Extras\Electronic Documentation
- **Mac OS**: Macintosh HD/Applications/FileMaker Developer 7/English Extras/Electronic Documentation

To view the PDF files, you need a PDF reader. In Mac OS X, you can use either the built-in Preview application or Adobe Reader. Windows users need Adobe Reader. If you do not have Adobe Reader, you can download it from the Adobe web site at www.adobe.com.

Important You can download PDFs of FileMaker 7 documentation from www.filemaker.com/downloads. Any updates to this document are also available from the web site.

All of the PDF files use the tagged Adobe Portable Document format (PDF). Tagged PDF files work with assistive technology such as the screen readers JAWS for Windows and Window-Eyes. For more information about tagged PDF files, see the Adobe web site at www.adobe.com.

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 Developer.

For information about technical support and customer service, see:

www.filemaker.com (North American customers)

www.filemaker.com/int (customers outside North America)

or choose Help menu > FileMaker on the Web. At the web site, you will find the service options available to North American customers, as well as links to FileMaker worldwide sites, answers to frequently asked questions, and access to the TechInfo Knowledge Base 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 American customers can also call (800) 965-9090 to learn about the service options available.

About the license key

FileMaker software comes with a unique, 35-character-string license key. Do not lose this license key; it cannot be replaced. We recommend that you keep the license key in a safe place in case the software ever needs to be reinstalled.

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

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

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

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 33.

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 Knowledge Base 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 Knowledge Base is available on the product support pages of the FileMaker web site at www.filemaker.com.

Chapter 2
Using FileMaker Developer features

In addition to all of the features that are available with FileMaker Pro, FileMaker Developer contains a number of advanced development and deployment features to speed up and enhance the development process. The FileMaker Developer features explained in this chapter are:

- the Database Design Report feature for publishing comprehensive documentation on the schemas of databases
- the Script Debugger for systematic testing and debugging of FileMaker scripts
- the File Maintenance features for compacting the size of database files and improving their performance
- the Custom Functions feature for creating custom functions for use anywhere within a file

FileMaker Developer also includes the Developer Utilities for creating, customizing, and deploying runtime database solutions. See "Using the Developer Utilities" on page 17.

Creating Database Design Reports

Use the Database Design Report feature to document the schema of your database and publish it to an HTML or XML file. You can choose which elements and database tables in the database that you want the report to cover. The HTML version of the report is hyperlinked and can be viewed or printed in a web browser.

With the Database Design Report feature you can:

- examine a textual representation of your database schema
- gather statistics on the structure of your database
- use the information in the report to recreate the structure of your database if you lose the original database files

A Database Design Report in HTML format includes a Report Overview that provides a snapshot of the elements in each database file. The Report Overview contains hyperlinks to details about all elements in each database file. A Database Design Report in XML format contains the same information, ready to be transformed into whatever format you require.

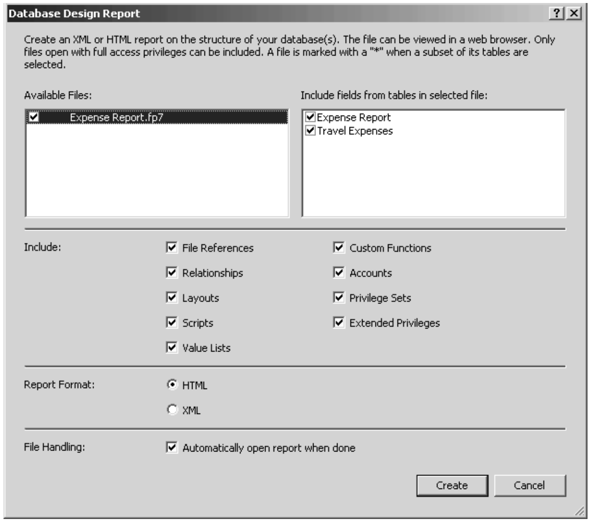
To create a Database Design Report:

1. Open all database files for which you want to produce a Database Design Report.

You must have full access privileges for any file for which you want to produce a Database Design Report and the file must be open in FileMaker Developer. You can run a Database Design Report on local or remote files.

2. Choose File menu > Database Design Report.

3. In the Available Files list, clear any files that you want to exclude from the report by clearing the checkbox associated with the file.



Database Design Report

Create an XML or HTML report on the structure of your database(s). The file can be viewed in a web browser. Only files open with full access privileges can be included. A file is marked with a "*" when a subset of its tables are selected.

Available Files:
☑ Expense Report.fp7

Include fields from tables in selected file:
☑ Expense Report
☑ Travel Expenses

Include:
☑ File References
☑ Relationships
☑ Layouts
☑ Scripts
☑ Value Lists
☑ Custom Functions
☑ Accounts
☑ Privilege Sets
☑ Extended Privileges

Report Format:
● HTML
○ XML

File Handling:
☑ Automatically open report when done

Database Design Report dialog box

4. If there are any files that contain tables that you want to exclude from the report, select the file in the Available Files list.

The tables in the file appear in the Include fields from tables in selected file list. You can then deselect any table in the list.

By default, all tables in all selected files are reported.

5. Clear elements that you want to exclude from the report.

By default, all elements in all selected files are reported. Each selected element, if present, will be reported on for each selected file.

6. If you do not want the report to be published in the default HTML format, select XML in the Report Format section.

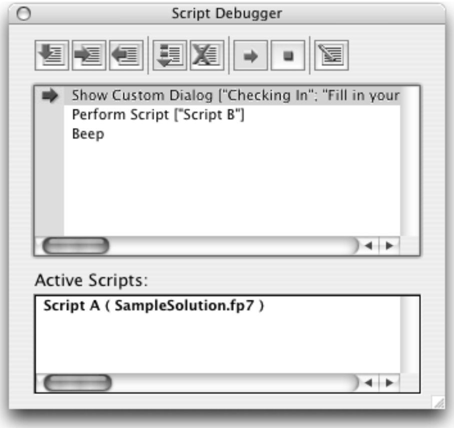
7. If you do not want the report to automatically open when done, clear the checkbox for this option in the File Handling section.

8. Click Create.

Using the Script Debugger

Use the Script Debugger to troubleshoot scripts in FileMaker databases. The Script Debugger enables you to execute your scripts step-by-step to view issues that may arise as the script is performed.

You control how the Script Debugger steps through the script either by using the buttons in the Script Debugger window or the commands in the Debug menu. If you find a step that requires modification, you can open ScriptMaker™ from a button in the Script Debugger window.



Script Debugger

```
Show Custom Dialog ["Checking In"; "Fill in your
Perform Script ["Script B"]
Beep
```

Active Scripts:

Script A (SampleSolution.fp7)

Script Debugger window

Tip To enable the Script Debugger from the Define Scripts dialog box, press Shift and click the Perform button. To disable the Script Debugger, press Ctrl (Windows) or Command (Mac OS) and click the Perform button. For more information on keyboard shortcuts, see Help.

The Script Debugger recognizes the privileges attached to each script. A script will only appear in the Script Debugger if you have editing privileges for the script and the access privileges for the script are set to Modifiable.

If a script with access privileges set to Modifiable performs a script with Executable Only access, the Executable Only script will perform in its entirety without showing its steps in the Script Debugger. If an Executable Only script performs a script with privileges set to Modifiable, only the steps in the Modifiable script will appear in the Script Debugger. For more information about script privileges and running scripts with full access, see Help.

To run scripts in debug mode, select Scripts menu > Debug Scripts.

Choose	To
Step	Execute the script one step at a time. If the script step is Perform Script, the Script Debugger will execute the sub-script and proceed to the next line of the calling script. The Script Debugger will execute all sub-script steps until it encounters a breakpoint.
Step Into	Execute the script one step at a time. If the script step is Perform Script, the Script Debugger will step to the first line of the sub-script and await user input before proceeding to the next sub-script step.
Step Out	Execute all script steps in the current script and, if the script is a sub-script, return to the line after the Perform Script step in the calling script. If the script is not a sub-script, the Step Out command will cause the Script Debugger to execute all script and sub-script steps until it encounters a breakpoint.
Run	Execute all script steps until a breakpoint is encountered.
Stop Execution	Halt execution of a script and close Script Debugger.
Set Next Step	Set the step execution pointer to the highlighted script step. This command passes control to the highlighted step but does not perform the step. The highlighted step is performed when script execution is resumed. Any script steps between the last executed step and the assigned next step are not executed. Click a step to highlight it, then click the Set Next Step button.
Set/Clear Breakpoint	Set or clear a breakpoint from the selected line. Breakpoints can be set or cleared from the Script Debugger or from ScriptMaker. Breakpoints are saved with the file that contains the script. Breakpoints are ignored by FileMaker Pro and when the Script Debugger is not in use. Breakpoints allow the Script Debugger to perform large sections of the script that do not need to be looked at closely.
Remove Breakpoints	Clear all the breakpoints from the current script. The Remove Breakpoints command is only in the Debug menu.
Go To ScriptMaker	Halts script execution and opens ScriptMaker so you can edit the script. The command to open ScriptMaker does not appear in the Debug menu.

Note In order to facilitate proper script debugging, the Script Debugger overrides some script steps. The Allow User Abort script step with the option set to off will not prevent you from stopping the execution of a script. The Adjust Window script step with the options of hide or minimize will not hide or minimize the window when encountered through the Step or Step Into buttons.

Using the File Maintenance feature

Use the File Maintenance feature to improve the performance of your database files. The Compact File option removes free disk space from a file, reducing the file size and streamlining data access. The Optimize File option improves file access speed by moving logically related data closer together. Both features reduce disk-head movement. Improvements will be especially noticeable when you are using large files and performing finds or sorts, or executing scripts that operate on all records.

The Compact File option compresses the file by removing free space, combining partially full disk pages and eliminating gaps in the data. This feature is useful in reducing the size of a database file. If there is going to be more data added to the file, you might not want to use the Compact File feature, as it will increase the amount of fragmentation occurring when new data is written to the database in the future. If you do add data to a file that has been compacted, you can defragment it with the Optimize File feature.

The Optimize File option defragments the file to make the physical arrangement of data match the logical arrangement. This feature is especially useful for increasing the speed at which finds and sorts are performed on files with a large number of records. You can use the Optimize File feature at any time, even on databases that will have data added to them.

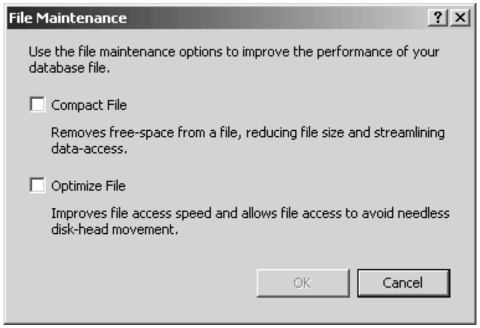
Both File Maintenance options work on the currently open database file, without creating another copy. If you are hosting files, networked clients will not have access to the file maintenance features for those files.

You can use the options together or by themselves. You can cancel the operation of the features at any time with the compression and defragmentation that has been performed up to the point of cancellation still retained.

1. Open a database file.

2. Choose File menu > File Maintenance.

3. Select one or both options, then click OK.



Use the file maintenance options to improve the performance of your database file.

□ Compact File

Removes free-space from a file, reducing file size and streamlining data-access.

□ Optimize File

Improves file access speed and allows file access to avoid needless disk-head movement.

File Maintenance dialog box

A File Maintenance Status dialog box shows you the progress of the features. An alert dialog box tells you when file maintenance is complete.

Using custom functions

Use the Custom Functions feature to create custom functions that can be reused anywhere in the database file in which they are created. Once formulas are written for the function, they don't have to be rewritten to be applied to other fields or used in other scripts.

You can maintain and edit custom functions and the formulas they contain in one central location. Any change made to the custom function will be copied to all instances where that custom function has been used.

- Any users of the database file that contains a custom function can use the function if they have the proper permissions.
- Custom functions will run in both FileMaker Pro and FileMaker Developer, and across multiple platforms.
- Newly defined functions appear under their own category in the functions list of the Specify Calculation and Define Custom Functions dialog boxes.
- If you do not wish to reveal your custom functions, you can disable the display of them in these dialog boxes.

- If you do not have full access privileges and attempt to edit a scripting or field calculation that uses a custom function that is restricted to full access accounts, you will see the string <Private Function> instead of the custom function. You cannot change the calculation when <Private Function> appears in the formula.

Creating custom functions

You must have Full Access privileges to the currently active database to use the Custom Functions feature.

To create a custom function:

1. Choose File menu > Define > Custom Functions.
2. In the Define Custom Functions dialog box, click New.
3. In the Edit Custom Function dialog box, for Function Name, type a name.

Custom function names have the following qualifications:

- The name must be unique
- The name cannot contain spaces
- Names cannot exceed 100 characters in length.
- Names cannot contain spaces; be sure to replace spaces with underscores or running the parts of the name together.

4. Build a formula.

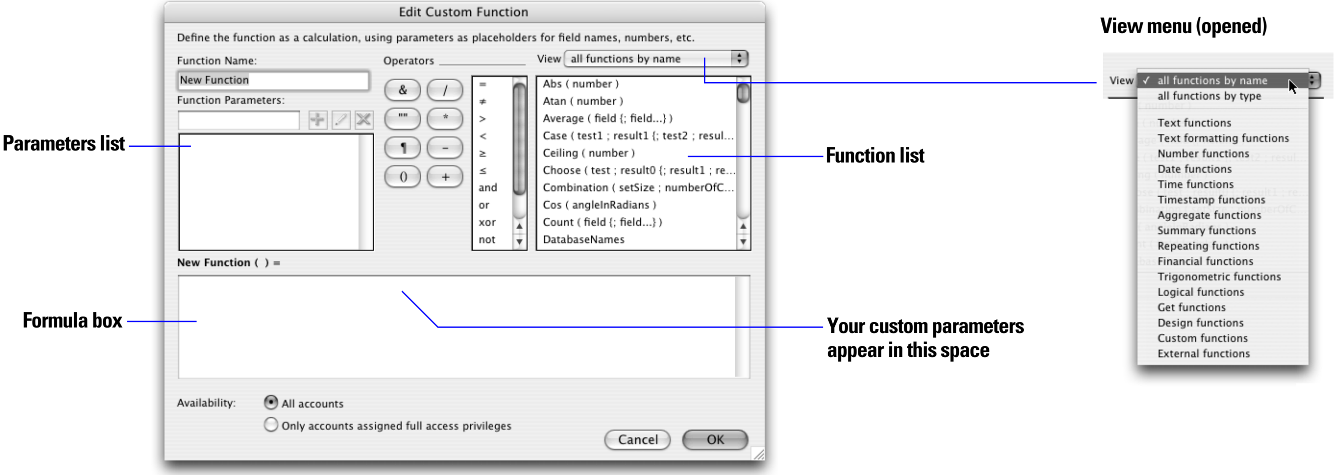
For more information about building formulas, see Help.

5. If the formula requires parameters, type the parameter name in Function Parameters and click the ✚ Plus button.

You can edit or delete a parameter name by selecting it in the list and clicking the ✎ Edit or ✖ Delete buttons.

6. Click where you want an item to appear in the formula box.
7. Perform one of the following actions.

To add a	Do this
Reference to a parameter	In the parameters list, double-click a parameter name.
Mathematical or text operator	In the keypad, click an operator.
Comparison or logical operator	For Operators, choose an operator from the list.



Edit Custom Function

Define the function as a calculation, using parameters as placeholders for field names, numbers, etc.

Function Name: New Function

View: all functions by name

New Function ( ) =

Availability: ● All accounts ○ Only accounts assigned full access privileges

View menu (opened)

Parameters list

Function list

Formula box

Your custom parameters appear in this space

Edit Custom Function dialog box

To add a	Do this
Constant value	Type the value.
Function	In the functions list, double-click a function. In the formula box, replace the example parameter with a value or expression.

You can also type parameter names, operators, and functions in the Edit Custom Functions dialog box instead of using the lists or keypad.

8. All accounts is the default option for Availability and allows all users of the current database to see and use the custom function.

9. If you want the custom function to be available only to those with full access privileges, select Only accounts assigned full access privileges.

10. Click OK.

Editing custom functions

The changes you make to a custom function are applied to that function wherever it is used.

To edit a custom function:

1. Choose File menu > Define > Custom Functions.

2. In the list of custom functions, select the function to be edited then click Edit.

3. In the Edit Custom Functions dialog box, make the needed changes, then click OK.

To change the way functions are sorted in the Define Custom Functions dialog box, choose a category from the View list.

1. In the Edit Custom Function dialog box, make the required changes.

2. Click OK.

3. Continue working with custom functions or click OK to close the Define Custom Functions dialog box.

Deleting custom functions

When you delete a custom function, it is no longer available for use.

If you have full access privileges to a database and are editing a calculation that uses a custom function that has been deleted, the name of the function is replaced with the string <Function Missing>.

To delete a custom function:

1. Choose File menu > Define > Custom Functions.

2. In the list of custom functions, select the function to be deleted and click Delete.

3. Continue working with custom functions or click OK.

Duplicating custom functions

You can make a copy of a custom function and then edit the copy to perform a similar operation.

To duplicate a custom function:

1. Choose File menu > Define > Custom Functions.

2. In the list of custom functions, select the function to be duplicated and click Duplicate.

The copy of the function appears in the list of custom function with "Copy" appended to its name.

3. Continue working with custom functions or click OK.

Chapter 3
Using the Developer Utilities

FileMaker Developer provides Developer Utilities that let you:

- bind your database files into a stand-alone runtime database solution that does not require FileMaker Pro or FileMaker Developer 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
- remove administrative access from all accounts and prevent users from modifying most design or structural elements of your databases
- add the FileMaker Pro filename extension to your files
- add your own logo or graphic to the closing splash screen of your stand-alone runtime database solution

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
- multiple privilege sets that can specify levels of access to layouts, menus, specific tables, record, fields, and so on
- password-protected accounts assigned to privilege sets that determine the level of access of account users

Note FileMaker Pro and FileMaker Developer now allow you to include as many database tables as you need in a database file. This capability eliminates one of the main reasons for using multiple files. However, other elements, like scripts and access privileges, are stored at the file level and so some complex solutions will still benefit from using multiple files.

Overview of preparing your solution files

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

The Developer Utilities are used to modify sets of database solution files. This chapter describes three main stages in the process:

1. To modify database solution files without creating a runtime database solution, see “Modifying database solution files” on page 18.

2. To prepare files for a runtime database solution and ensure that their file references will work in the solution, see “Considerations for a runtime database solution” on page 20.

3. To create a runtime database solution, see “Binding databases into runtime database solutions” on page 22.

Here are some general considerations for preparing solution files:

- If desired, create a custom theme for all the layouts in your database solution. See "Creating custom layout themes" on page 47 for information.
- If you're creating a solution that will have versions for Windows and Mac OS X, test the different versions of the solution on their respective platforms. For more information, see Help.
- If you have used multiple files instead of multiple tables in a single file, all files for your solution should be in the same folder before being bound into a runtime solution. If it is not practical to keep all files in one folder, be sure to include a file reference to each file that is just the filename. See "Checking file references" on page 21.
- Make sure to specify every file that's related to the solution, so that if you modify filenames all file references will be updated.
- If you have used multiple database files, decide which file will be the primary file that users open first. The primary file stores the custom settings. 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.
- Create scripts and buttons for users to navigate from the primary file to auxiliary files and layouts in the solution. See "Using scripts to control your solution" on page 40.
- Create documentation about your database solution. See "Providing user documentation" on page 33.
- Although the Developer Utilities use a 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.

Modifying database solution files

Use the Developer Utilities to produce a new set of database files, to customize them, or to create a runtime database solution.

Note You must convert database files from versions of FileMaker Pro earlier than 7.0 before you can use them with the Developer Utilities. For more information, see the *Converting FileMaker Databases from Previous Versions* guide.

To use the Developer Utilities:

1. Close all of your database files that you are going to customize.
2. Choose File menu > Developer Utilities.
3. If you have used the Developer Utilities on the same database solution before and saved your settings, click Load Settings.

A Select a file dialog box opens so that you can browse to find your settings file. See "Saving and reusing Developer Utilities settings" on page 25.

4. Click Add to locate the files that you want to customize.
5. When you have added all the files that you want to customize, do one or more of the following:

To	Do this
Select the primary file	Double-click the file in the list.
Rename a file	Select the file in the list, type the new name in the Rename file box, and click Change. **Note** Do not type a filename extension. See "Choosing filenames for runtime database solutions" on page 23.
Remove a file	Select the file in the list and click Remove.

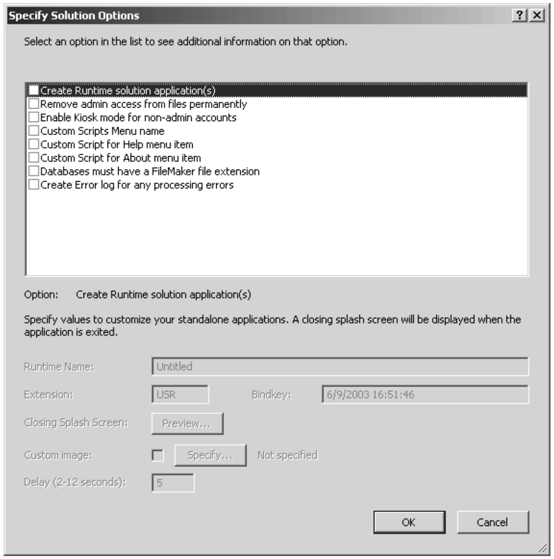
6. Under Project Folder, click Specify to choose the location in which the copy of the database solution will be saved.
7. In the Choose a folder for this project's files dialog box, select or create a folder and click OK (Windows) or Choose (Mac OS X).

8. If you do not want the new files to overwrite earlier versions of them, clear the Overwrite matching files within the Project Folder checkbox.

Important If Overwrite matching files within the Project Folder is selected, the Developer Utilities will overwrite files with the same names as those in the list of files.

9. Do one of the following:

- To customize your database files or bind the files to a runtime solution, under Solution Options click Specify.



Specify Solution Options

Select an option in the list to see additional information on that option.

- □ Create Runtime solution application(s)
- □ Remove admin access from files permanently
- □ Enable Kiosk mode for non-admin accounts
- □ Custom Scripts Menu name
- □ Custom Script for Help menu item
- □ Custom Script for About menu item
- □ Databases must have a FileMaker file extension
- □ Create Error log for any processing errors

Option: Create Runtime solution application(s)

Specify values to customize your standalone applications. A closing splash screen will be displayed when the application is exited.

Runtime Name: Untitled

Extension: USR Bindkey: 6/9/2003 16:51:46

Closing Splash Screen: Preview...

Custom image: □ Specify... Not specified

Delay (2-12 seconds): 5

Specify Solution Options dialog box

- To create a copy of your database files with new names, skip to step 12.

10. In the Specify Solution Options dialog box, select one or more options.

To	Do this
Bind databases to runtime applications	Choose Create Runtime solution application(s) See “Binding databases into runtime database solutions” on page 22. **Note** This option can be combined with all others, except Databases must have a FileMaker file extension.
Permanently prohibit any administrative access to your solution	Choose Remove admin access from files permanently See “Removing full access privileges from databases” on page 27. **Important** Once removed, administrative access cannot be restored to the custom solution.
Force accounts without full access privileges to open your solution in Kiosk mode	Choose Enable Kiosk mode for non-admin accounts See “About Kiosk mode” on page 39.
Give the Scripts menu in your solution a custom name	Choose Custom Scripts Menu name See “Customizing About, Help, and Scripts menus” on page 43.
Run a specified script from the Help menu command	Choose Custom Script for Help menu item See “Customizing About, Help, and Scripts menus” on page 43. **Note** This option is only available if there are scripts in the primary file.
Run a specified script from the About menu command	Choose Custom Script for About menu item See “Customizing About, Help, and Scripts menus” on page 43. **Note** This option is only available if there are scripts in the primary file.

To	Do this
Add the FileMaker extension to the filenames of database files	Choose Databases must have a FileMaker file extension **Note** This option is not available if you select Create Runtime solution application(s). You can use this feature to add extensions to files that do not have extensions.
Create a log file to record any errors encountered during processing	Choose Create Error log for any processing errors See "Creating an error log" in the next section.

11. Click OK.

12. To be able to quickly repeat the process, click Save Settings, and choose a folder and location for your settings file.

See "Saving and reusing Developer Utilities settings" on page 25.

13. Click Create.

The Developer Utilities copy all the selected database files, with the modifications made by the Utilities, to the Project Folder. If the default option to overwrite the destination files has been left on, the specified Project Folder will be used to store the files copied by the Developer Utilities. If the default option to overwrite the files has been cleared, the specified Project Folder will be used if it is empty or a new one based on the name of the Project Folder will be created.

Creating an error log

Some errors do not trigger error messages. An error log will capture more detail on any errors encountered during processing.

To create an error log:

1. In the Specify Solution Options dialog box, select the Create Error log for any processing errors.

2. Click Specify.

If you don't specify a filename and location for the error log, it will be saved to the project folder with the filename Logfile.txt.

3. Use the Specify a file dialog box to select a location and provide a filename for the error log.

If an error occurs during the processing of the options, the error is logged in the error log. An error message may also indicate that an error has been encountered.

Considerations for a runtime database solution

You should address the following issues before binding the files to the runtime application.

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

In the runtime application, there are no menu options to open or close files. If you want users to open and close auxiliary files, you must provide scripts in your solution to perform these operations. In each auxiliary file you must place a button or startup script that returns users to the primary file.

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

Unless your runtime database solution files are password protected, users can open and modify the files in FileMaker Pro or FileMaker Developer. You can also make your files permanently unmodifiable. See "Removing full access privileges from databases" on page 27.

- **Will this be a runtime database solution used on both platforms?**

See "Binding files for both platforms" on page 22.

- **How will you provide updates for your users?**

You can make it easier for users to update your runtime database solution files by providing scripts in your primary file to export their data and import it into the updated solution. See "Importing data into upgraded runtime solutions" on page 26.

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

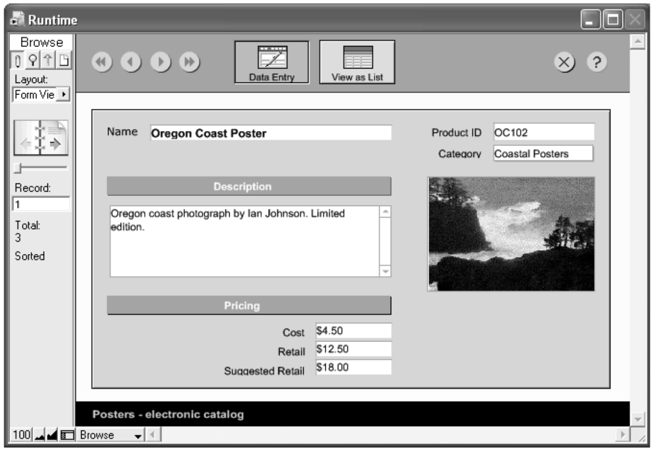
It's a good idea to set document margins if your runtime database solution will be printed from a variety of printers. For more information, see Help.

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

You can change the main spelling dictionary language for your database solution by choosing one from the dictionaries supplied with FileMaker Developer. Your users can add or modify a user-defined dictionary to the runtime application. For more information, see Help.

- **Are your original database solution files in more than one folder?**

See "Checking file references" in the next section.



Record: 1

Total: 3

Sorted

Name: Oregon Coast Poster

Product ID: OC102

Category: Coastal Posters

Oregon coast photograph by Ian Johnson. Limited edition.

Cost: $4.50

Retail: $12.50

Suggested Retail: $18.00

Posters - electronic catalog

Primary file of a sample runtime solution

Checking file references

A file reference stores the path or paths that the runtime application searches to access an external table, script, or value list. Each file reference can consist of one or more paths, separated by carriage returns. File paths are searched in the order in which they appear. The runtime application opens the first file it is able to locate. For more information, see Help.

Tip You may want to put multiple tables in one file and so avoid any potential difficulties with multiple files.

During the development of a database with multiple files, you may want to have some of the files in separate folders. During the creation of a runtime database solution, however, all files are moved into the same folder as the runtime application. Your file references should take this into consideration, whether your solution starts out in one folder or not.

For this reason, every file reference must include a path that is just the filename of the file being referenced. Although the runtime application will check other file references, it will then be able to find the file in the same folder in which it resides. You can still keep any absolute or relative paths in the same file reference in case the files are also used in FileMaker Pro or FileMaker Developer.

To check file references:

1. Choose File menu > Define > File References.

The Define File References dialog box lists all the files references in the current database. Check to ensure that for each file named, there is a reference in the File Path List that is to the filename only, without any folders. The reference will look like this: `file:MyFile.fp7`.

2. If any file does not have a reference to the filename only, click New in the Define File References dialog box.
3. In the File Path List, type the filename, including the extension.
4. For File Reference Name, type a name. This is the name that will appear in all lists that display file references.
5. Click OK to save the file reference.

Binding files for both platforms

If your solution will be used in Windows, bind it using the Developer Utilities for Windows. If your solution will be used on Mac OS X, bind it using the Developer Utilities for Mac OS X. If you’re creating a solution to be used on both Windows and the Mac OS X, create two separate runtime solutions by binding the original solution files twice: first using FileMaker Developer Utilities for Windows, and then using FileMaker Developer Utilities for Mac OS X. Use the same binding key on both platforms. Also, remember that binding keys are case-sensitive.

Binding databases into runtime database solutions

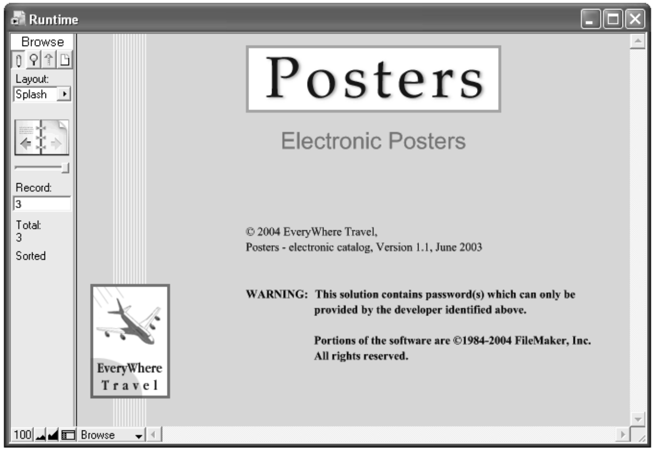
Use the Developer Utilities to produce a stand-alone runtime database solution that users can access without running FileMaker Pro or FileMaker Developer. The Developer Utilities create a copy of your files, and bind the database file or files to a runtime application with a name that you specify.

None of the commands on the Define submenu opened from the File menu are available in the runtime application. See “Feature comparison of the runtime application with FileMaker Pro” on page 71. The FileMaker Developer features are also stripped from the runtime application.

A runtime database can, however, be opened in either FileMaker Pro or FileMaker Developer. The full functionality of these applications will be enabled, except if full access privileges have been removed. See “Removing full access privileges from databases” on page 27.

Runtime database solutions cannot be published over a network, the Internet, or an intranet unless you use FileMaker Pro or FileMaker Developer instead of the runtime application. For a complete list of the differences between the runtime application and FileMaker Pro, see “Feature comparison of the runtime application with FileMaker Pro” on page 71.

You may need to bind your database files several times before you prepare them for delivery to your users. When you have completed development and the final version is bound and ready to distribute, you should thoroughly test your runtime solution to ensure that it behaves as expected. See “Considerations for a runtime database solution” on page 20.



© 2004 EveryWhere Travel,
Posters - electronic catalog, Version 1.1, June 2003

WARNING: This solution contains password(s) which can only be provided by the developer identified above.

Portions of the software are ©1984-2004 FileMaker, Inc. All rights reserved.

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

For information about what users need to use your runtime database solution, see Chapter 4, “Distributing runtime database solutions” on page 29.

To bind database files into a runtime database solution:

1. Follow steps 1 to 9 in “Modifying database solution files” on page 18.

2. In the Specify Solution Options dialog box, select Create Runtime solution application(s).

3. For Runtime Name, type a name for your runtime application.

The runtime name is used for the runtime application filename and for the name of the folder that contains the runtime database solution files. See “Choosing filenames for runtime database solutions” in the next section.

4. For Extension, type a three-character filename extension.

The extension is used to associate the solution files with the runtime applications. See “Assigning three-character filename extensions” on page 23.

5. For Bindkey, type a key between 1 and 24 characters long.

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. See “Setting the binding key” on page 25.

Important Binding installs system files pertaining to each platform.

6. To add a company logo or other custom image to the closing splash screen, click Specify, select the closing image, and click Select.

The image should be at least 382 x 175 pixels (72 dpi) or higher, otherwise it will be distorted when displayed. The supported image formats are JPEG and GIF.

7. For Delay, set the number of seconds that you want the splash screen to display.

You can preview the effect that your custom splash screen will have by clicking the Preview button.

8. Once you have specified options, click OK.

9. To be able to quickly repeat the process, click Save Settings, and choose a folder and location for your settings file.

See “Saving and reusing Developer Utilities settings” on page 25.

10. Click Create.

The Developer Utilities copy all of the runtime files to a new folder, created inside the Project Folder and named after the runtime solution. For information on bundling the necessary files and delivering your runtime database solution to your users, see “Distributing runtime database solutions” on page 29.

Choosing filenames for runtime database solutions

When choosing filenames for runtime database solutions, consider the platforms on which your runtime solution will be used so your scripts and lookups will work properly. The operating system limits the number of characters supported for filenames.

Platform	Filename support
Mac OS X	255 characters
Windows 2000, XP	255 characters

Windows filenames must not start with a space. For cross-platform compatibility, do not use the following characters in filenames: quotation mark ("), slash (/), backslash (\), angle brackets (<>), asterisk (*), questions mark (?), colon (:), vertical bar (|).

Assigning three-character filename extensions

The three-character filename extension associates all of the runtime solution files with the 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. Use an extension that is unique to your users’ computer systems.

The Developer Utilities automatically update all files to use the three-character extension that you specify and append the extension to the filenames. Internal file references used in relationships, scripts, and external value lists are updated to interact with the new filenames.

Note No matter what the filename extensions are, runtime database files can still be opened in FileMaker Pro and FileMaker Developer. To prevent users from modifying your runtime database solutions, create passwords for specific access privileges or select the Remove admin access from files permanently option in the Developer Utilities before you bind the files into a runtime database solution. See "Removing full access privileges from databases" on page 27.

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 Utilities append the extension to all database filenames in the runtime database solution during the binding process.

Assigning the extension for Mac OS X solutions

In the Mac OS X, the three-character extension becomes the creator code for the runtime application. The creator code must be unique to ensure that the Mac OS X 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 Utilities insert 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, Inc. 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 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 in Windows

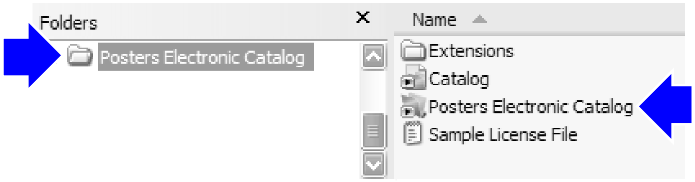
If the three-character extension is not unique, it might cause registry conflicts in Windows. For example, if you use the .fp7 extension for your runtime database solution and your users have FileMaker Pro installed on their hard disks, all of their FileMaker Pro document icons will change to the runtime icons. Additionally, FileMaker Pro documents will no longer automatically open the FileMaker Pro application.

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.

Setting the binding key

The runtime name that you specify in the Developer Utilities 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 Developer Utilities use 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 solution for both platforms, use the same key when you bind the solution in Windows and in the Mac OS X. Also, keep in mind that the binding key is case-sensitive on both Windows and Mac OS X machines.

Note Use a binding key you'll remember and record it in a safe place. (You can do this by saving the Developer Utilities settings file: see "Saving and reusing Developer Utilities settings" on page 25.) 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.

Modifying bound runtime files

You can open a bound runtime file in FileMaker Pro and FileMaker Developer to make modifications to it. However, if you selected the Remove admin access from files permanently option when you bound the files, then you can't regain access to the Define Database, Define Value Lists, Define File References, Define Accounts & Privileges, and Define Custom Functions dialog boxes, or the Layout Mode and ScriptMaker menu commands. In this case, you'll have to open the original database files in order to make design changes in FileMaker Developer and then rebind them using the binding key that you assigned to that runtime database solution.

See "Distributing updates to runtime database solutions" on page 37.

Saving and reusing Developer Utilities settings

You can save the settings you specify in the Developer Utilities dialog box and reuse them again. The settings include the list of files to be modified. After you have entered your specifications in the Developer Utilities, click Save Settings. Choose a location and filename for the settings file. The extension .sav is automatically added and should not be changed. You can save as many settings files as you want, using different locations and names.

Use the Developer Utilities settings file to repeat the same processes on your database files with the Developer Utilities. When you open the Developer Utilities, click Load Settings, locate the settings file, and click Load.

Modifying a set of solution settings

To modify a settings file:

1. Open the Developer Utilities.
2. Click Load Settings.
3. Locate and select the settings file that you want to modify.
4. Click Load.
5. Specify options.
6. Click Save Settings, then click Save.
7. Click Yes to replace the file.

Your alterations are saved and may be reused again.

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, the Binder utility in the FileMaker Pro 4.0 Developer Edition, or the Developer Tool in FileMaker Developer 5.x and 6.0, you can upgrade your solution and provide your users with the converted files. Files bound to a runtime application using the earlier tools must be rebound using the Developer Utilities.

You must convert FileMaker Pro files created in earlier versions to the new file format. You can convert a single file or convert multiple files at once. For more information about converting files, see the *FileMaker Pro User's Guide* and the *Converting FileMaker Databases from Previous Versions* guide.

Once you have converted the files, you can upgrade them to take advantage of newer FileMaker Pro and FileMaker Developer features. If necessary, create scripts to import users' existing data from the old runtime database solution into the new, upgraded solution. Use the Developer Utilities to bind the solution files into a new, upgraded runtime database solution. See "Binding databases into runtime database solutions" on page 22.

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

Importing data into upgraded runtime solutions

You can include scripts in the new runtime database solution files that allow users to import records from the old runtime files. The old files must first be converted to the new file format.

Note When creating your upgraded runtime solution, use a different extension than the one you used for your old runtime solution.

To prepare your upgraded solution for importing data:

1. Create a folder named "Old Solution Files" inside the folder that contains the new runtime solution database files.
2. Place copies of the old runtime solution database files in the "Old Solution Files" folder.
3. In each upgraded file, create a script to convert the old solution file and import records from it to the new file.

The functionality of the script should include:

- Convert File [<filename of the old version of the solution file>]
- Import Records [<old filename with the new solution's extension>]
- Close File [<old filename with the new solution's extension>]

4. Add a button to activate the script.

5. Repeat steps 3 and 4 for each upgraded file.

6. Use the Developer Utilities to bind your upgraded solution files into the new runtime database solution.

7. Test your buttons in the runtime application.

Use sample data to make sure the records import properly and data is imported to the correct fields.

8. Distribute the new solution files that contain the buttons.

9. Provide instructions telling users how to import data into the new solution files.

Users can copy their old files into the "Old Solution Files" folder and use the buttons in the new files to convert the old files and import records from them into the new solution files.

Removing full access privileges from databases

FileMaker Pro and FileMaker Developer use accounts, privilege sets, and extended privileges to protect FileMaker databases. For more information, see Help. You can use the Developer Utilities to remove all administrative accounts from a file. For more information about accounts and privileges, see Help.

Use the Developer Utilities to prevent users from altering the design and structure of your database files and from changing any accounts or privileges that you've set up. The Remove admin access from files permanently option deletes from the database all accounts that were using the Full Access privilege set. All Define dialog boxes will also be unavailable, except for the Extended Privileges tab of the Accounts & Privileges dialog box.

Note All database files must have at least one active account or they can't be opened. You can't use the Remove admin access from files permanently option if the accounts with Full Access privilege sets are the only active accounts in the file. If you attempt to do so, you will receive an error message.

Important Selecting this option *permanently deletes* from the database all accounts that were using the Full Access privilege set. This will permanently eliminate access to Layout mode and ScriptMaker, and all Define dialog boxes, except for the Extended Privileges tab of the Accounts & Privileges dialog box, for all database files in the solution, whether they're opened in a runtime application, in FileMaker Pro, or in FileMaker Developer. Structural and design elements of the files cannot be modified by anyone, including FileMaker employees. The only way to modify the tables, field definitions, relationships, scripts, or access privileges is by returning to the original file before it was customized by the Developer Utilities.

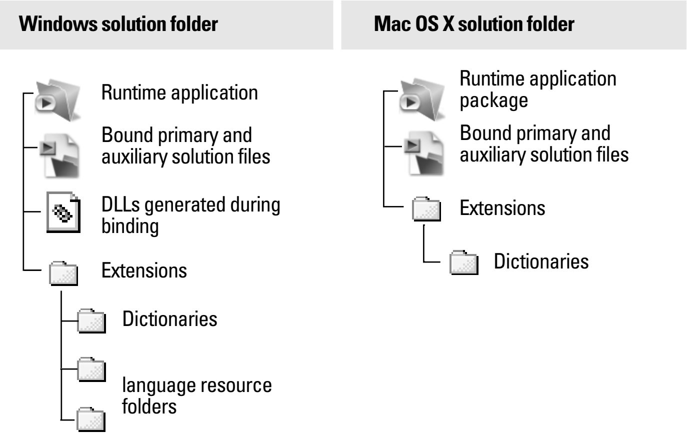
Consider the long-term needs of your users when defining access privileges. Communicate their access privileges to them clearly in the About layout and follow the rules specified by FileMaker. For more information, see "Adding custom scripts to the About menu command" on page 43.

Chapter 4
Distributing runtime database solutions

The final steps in developing your 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 or over a network—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 solution components

When you bind your database files into a runtime database solution, the Developer Utilities create a new solution folder and place the runtime application, the bound primary and auxiliary database files, and an Extensions folder inside it. For Windows runtime solutions there are also required Dynamic Link Library (DLL) files.


Windows solution folder
Mac OS X solution folder
Runtime application
Bound primary and auxiliary solution files
DLLs generated during binding
Extensions
Dictionaries
language resource folders
Runtime application package
Bound primary and auxiliary solution files
Extensions
Dictionaries

Example of Windows and Mac OS X solution contents for distribution

Important These files and folders must not be renamed.

If your runtime database solution requires custom files, you should provide the files with the runtime files. Plug-ins should be stored in the Extensions folder. If a developer uses a font not found on a user's system, the runtime application will make a font substitution. If a font is included with the runtime, provision should be made for its installation through the installer program. See "Using a custom installation program" on page 31.

In addition to the runtime files, you will need to provide installation instructions for your users. See "Documenting the installation procedures" on page 32.

Mac OS X runtime application package

The Developer Utilities generate a runtime application package for Mac OS X. The package will have the name that you give to your runtime solution. The runtime application package contains the core resources and code required for running the application. The contents of the package should not be altered, except to remove the language resource files. For more information, "Reducing solution size" on page 30.

The package contains a Contents folder, which has information on the package and the Frameworks, MacOS, and Resources folders. The Resources folder includes the language resources folders (*.lproj). The package can be opened by using the Show Package Contents command.

The Extensions folder accompanies the Mac OS X package and contains the Dictionaries folder. The Dictionaries folder contains dictionary files (*.mpr) for all of the languages supported. The Dictionaries folder also contains a file for the user spelling dictionary (User.upr).

DLL files for Windows runtime solutions

The Developer Utilities generate a set of Dynamic Link Libraries (DLLs) during the binding process. These are in the solution folder along with the runtime application and bound database files. They must be delivered as part of the runtime solution. If any of these files are missing or become damaged, they must be replaced.

DBConverter.dll
DBEngine.dll
DOMSupport.dll
FML10.dll
FMLayout.dll
FMOLE.dll
FMRSRC.dll
FMScript.dll
FMUserModel.dll
FMWrapper.dll
GdiPlus.dll
HBAM.dll
libeay32.dll
mfc70u.dll
MFCX.dll
msvcp70.dll
msvcp70d.dll
msvcr70.dll
msvcr70d.dll
NSViews.dll
omniDynamic400_rt.dll
omniORB400_rt.dll
omnithread30_rt.dll
PlatformSupport.dll
ProofReader.dll
ssleay32.dll
Support.dll
XalanDOM.dll
XalanExtensions.dll
XalanSourceTree.dll
XalanTransformer.dll
XCore.dll
XDraw.dll
xerces.dll
XercesParserLiaison.dll
XFC.dll
XGrfx.dll
XMLEngine.dll
XMLSupport.dll
XPath.dll
XSLT.dll
XText.dll

Extensions folder for Windows solutions

The Extensions folder of Windows runtime solutions contains a folder for each language supported and a dictionaries folder. Each language resource folder contains a DLL file for the language (FMRSRC.dll), a file containing the text for the interface, dialog boxes, and error messages (FMStrs.dls), and a file containing the Windows language ID for the language (lang.dat).

The Extensions folder also contains the Dictionaries folder. The Dictionaries folder contains dictionary files (*.mpr) for all of the languages supported. The Dictionaries folder also contains a file for the user spelling dictionary (User.upr).

Choosing a 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 your bundled solution on a CD-ROM, over a network, or via the Internet.

Reducing solution size

If you are planning on distributing your runtime solution via the Internet, you may want to reduce the size of the solution before you package the solution. Deleting dictionaries and language resource files for languages not supported by your database will reduce the size of the application.

To reduce the size of your solution:

Windows

1. Open the Extensions folder that is copied to the runtime solution folder.
2. Delete language resource folders for languages not supported by your solution.

Important Do not delete the language resource folder for English.

3. Open the Dictionaries folder.
4. Delete dictionaries for languages not supported by your solution.

Mac OS X

1. Open the Extensions folder that is copied to the runtime solution folder.
2. Open the Dictionaries folder.
3. Delete dictionaries for languages not supported by your solution.
4. Control-click the runtime application package and choose Show Package Contents.
5. Locate the Resources folder in the Contents folder.
6. Delete language resource folders (*.lproj) for languages not supported by your solution.

Using a custom installation program

You should use a custom installation program to package your runtime solution for installation by users. Configuring a custom installation application to automatically install runtime database solution files may require more engineering than using a compression utility, but will help to ensure that your users do not have difficulties installing your runtime solution.

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

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

Using a compression utility program

If your runtime database solution is not complex and you have confidence in the technical experience of your end users, you might consider a compression utility program rather than a custom installation program. To compress files, use a utility such as:

- WinZip for Windows (Windows 2000, XP) by WinZip Computing, Inc. (www.winzip.com)
- StuffIt Deluxe (Windows and Mac OS X) by Aladdin Systems (www.aladdinsys.com)

Sharing solutions over a network

Users cannot share your runtime database solution over a network unless they access the files using FileMaker Pro or FileMaker Developer installed on their machines. You must have a master password to enable or change network access to the file. For optimal performance, you 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 web site at www.filemaker.com.

What your users need

In order to run your runtime database solution, your users will need the same minimum equipment and software required by the FileMaker Developer application (see the *Installation and New Features Guide for FileMaker Pro and FileMaker Developer*). In addition, your users will need instructions for installing and starting your solution, and information about how to recover damaged files.

Documenting the installation procedures

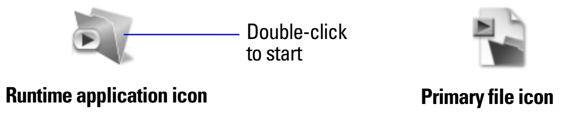
You'll need to provide instructions to your users on how to install your runtime database solution. Here's a list of things you should document:

- Provide written instructions for copying or installing your solution to your users' hard disks.
- 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, "Providing user documentation" on page 33.

Starting runtime database solutions

After a user has run the installation program for your runtime application, the files are installed on the user's hard drive. The solution's three-character filename extension is registered with the operating system (Windows) or in the system preferences (Mac OS X). This registration allows the operating system to locate and launch the runtime solution if the user double-clicks the primary or auxiliary solution files. 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 their 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.

Note When you make a change to your solution, 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, "Importing data into upgraded runtime solutions" on page 26.

Recovering damaged files

Power failures, hardware problems, or other factors can damage a FileMaker database file. If your database solution becomes damaged, your users will 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. Even if the dialog box does not appear, files can become corrupted and exhibit erratic behavior.

Once you know which file is damaged, you can recover it using the Recover command, if you have FileMaker Pro or FileMaker Developer installed. If your user only has the runtime application, 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 Open Damaged File dialog box.
- On Mac OS X machines, press Option + ⌘ while double-clicking the runtime application icon. Hold the keys down until you see the Open Damaged File dialog box.

During the recovery process, the 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 documentation, you should tell your users what to do after a file has been recovered. Tell your users to:

1. Recover the damaged solution file using the method described above for the type of computer they are using.
2. Open the recovered solution file in the runtime application.
3. Choose File menu > Save a Copy As.
4. In the dialog box, choose compacted copy (smaller) from the Save a (Windows) or Type (Mac OS X) drop-down list, name the file, and click Save.

   Give the compacted file the same filename as the original file.
5. Make a copy of the original database and import the data from the recovered file into it.

Providing user documentation

You should provide documentation for your database solution, whether it is a database that must be opened in FileMaker Pro or FileMaker Developer, or a standalone runtime database solution. There are several ways that you can provide documentation for your solution, including a printed manual, an online Help system, and an About layout that is available from any layout in the solution.

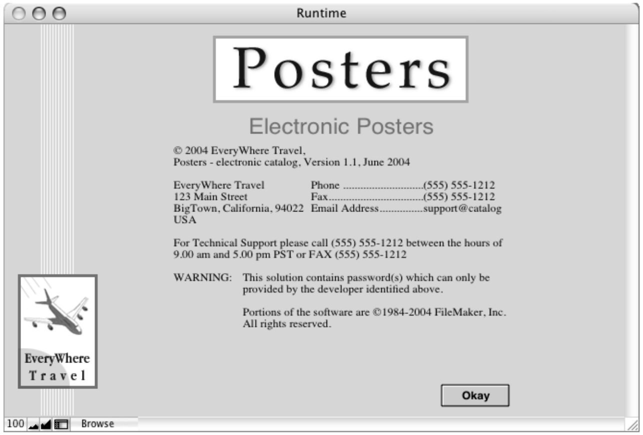
Create custom About and Help layouts that document what your database solution is, how to use it, and where users can go for more information. Then use the Developer Utilities to attach scripts to menu commands that open the About and Help layouts.

Note The FileMaker Pro and FileMaker Developer Help system is not available in runtime applications. However, Status bar Help (Windows) and Help tags (Mac OS X) are available.

Creating an About layout

For runtime database solutions, the FileMaker Developer license specifies that you must create an About layout that provides information for your users on how to contact you for technical support. FileMaker uses the About layout to distinguish databases created by developers using FileMaker Developer rather than users of FileMaker Pro.

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 35.



Electronic Posters

© 2004 EveryWhere Travel,
Posters - electronic catalog, Version 1.1, June 2004

EveryWhere Travel   Phone ....(555) 555-1212
123 Main Street   Fax....................................(555) 555-1212
BigTown, California, 94022   Email Address...............support@catalog
USA

For Technical Support please call (555) 555-1212 between the hours of 9.00 am and 5.00 pm PST or FAX (555) 555-1212

WARNING: This solution contains password(s) which can only be provided by the developer identified above.

Portions of the software are ©1984-2004 FileMaker, Inc. All rights reserved.

Example of an About layout

To create an About layout:

1. Choose View menu > Layout Mode.
2. Choose Layouts menu > New Layout/Report.

3. Type `About <your solution>` in the Layout Name box.

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 35.

4. Select Blank Layout.

5. Click Finish.

6. Include in the layout your logo, other graphics, and your company information.

7. Include text that notifies users if the solution files are protected with passwords or if full access privileges have been removed.

See "Your responsibilities as a developer" on page 35 for the exact legal wording.

8. Create a button that lets your users return to the main layout of the primary file.

9. Choose Scripts menu > ScriptMaker and create a script that goes to the About layout.

- Include the word "About" in the script's name.

10. Use the Developer Utilities to create a menu command that displays the About layout.

The Developer Utilities use the script's name for the name of the menu command. "Adding custom scripts to the About menu command" on page 43.

To display the About layout when the runtime application is started:

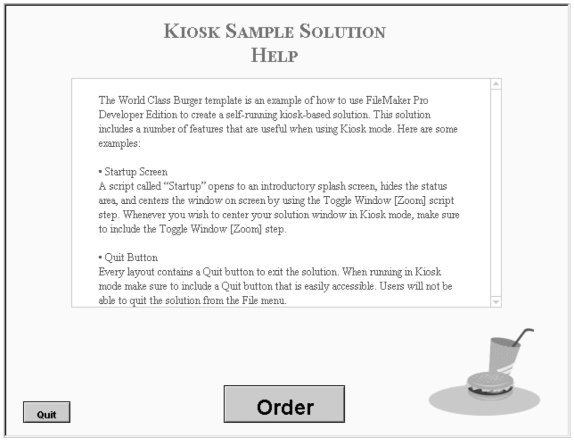
1. Open the primary file for the database solution.

2. Choose File menu > File Options > Open/Close.

3. Select Switch to layout and choose the About layout from the drop-down list.

Creating a custom Help layout

Create a Help layout that provides instructions for how to use your custom solution and add data to it. Then create a script in the primary file of your solution to display the Help system. Use the Developer Utilities to make the script available as a command in the Help menu. See "Adding custom scripts to the Help menu command" on page 45.



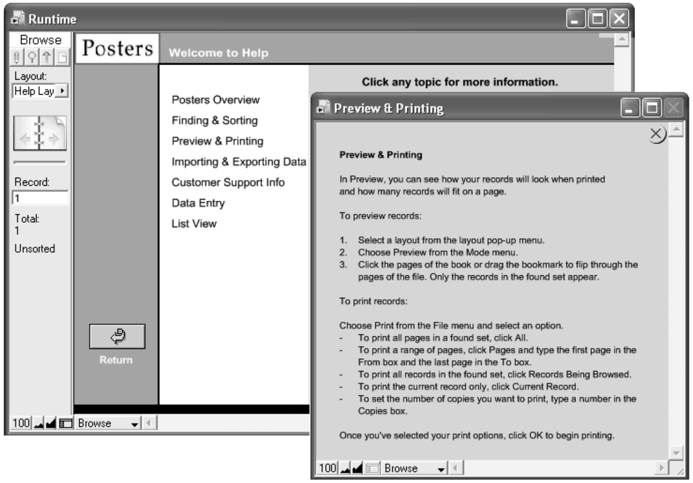
KIOSK SAMPLE SOLUTION
HELP

The World Class Burger template is an example of how to use FileMaker Pro Developer Edition to create a self-running kiosk-based solution. This solution includes a number of features that are useful when using Kiosk mode. Here are some examples:

▪ Startup Screen
A script called "Startup" opens to an introductory splash screen, hides the status area, and centers the window on screen by using the Toggle Window [Zoom] script step. Whenever you wish to center your solution window in Kiosk mode, make sure to include the Toggle Window [Zoom] step.

▪ Quit Button
Every layout contains a Quit button to exit the solution. When running in Kiosk mode make sure to include a Quit button that is easily accessible. Users will not be able to quit the solution from the File menu.

A kiosk example containing a single Help layout



Click any topic for more information.

Posters Overview
Finding & Sorting
Preview & Printing
Importing & Exporting Data
Customer Support Info
Data Entry
List View

Preview & Printing

In Preview, you can see how your records will look when printed and how many records will fit on a page.

To preview records:

1. Select a layout from the layout pop-up menu.
2. Choose Preview from the Mode menu.
3. Click the pages of the book or drag the bookmark to flip through the pages of the file. Only the records in the found set appear.

To print records:

Choose Print from the File menu and select an option.
- To print all pages in a found set, click All.
- To print a range of pages, click Pages and type the first page in the From box and the last page in the To box.
- To print all records in the found set, click Records Being Browsed.
- To print the current record only, click Current Record.
- To set the number of copies you want to print, type a number in the Copies box.

Once you've selected your print options, click OK to begin printing.

A runtime solution example with a separate Help window

Including printed documentation

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.

Other items appropriate for printed documentation include:

- how to install your bundled solution
- how to install custom files
- how to upgrade to new solution files
- how to use your Help system
- how to start the database solution (see "Starting runtime database solutions" on page 32)
- what to do in case of a damaged file (see "Recovering damaged files" on page 32)
- 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, see Help.

Important In the event that your runtime database solution files become damaged, make sure users have access to your technical support email 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.

Your responsibilities as a developer

FileMaker has established procedures for repairing files. If a customer complies with these procedures, then FileMaker may supply a repaired file to the customer.

If you distribute database files with passwords or you have removed full access privileges and do not want FileMaker to repair a file for a customer who requests this service, you must:

1. Notify your customers in writing and keep a record of such notice that your database solution contains passwords 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 custom scripts to the About menu command" on page 43 and "Creating an About layout" on page 33.
3. The layout name must begin with the word "About."
4. The About layout must contain these items:
   - solution name
   - your company name and contact information
   - your support policy (for example, how and when you are available for technical support)
5. 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 full access privileges have been permanently removed from your database solution by selecting the Remove admin access from files permanently option in the Developer Utilities, 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 accounts and privileges protection in a FileMaker file should not be viewed as an absolute barrier that will prevent a customer from accessing files. FileMaker cannot guarantee that a customer will not be able to identify or bypass the password through third-party solutions or tools. Therefore, FileMaker recommends that you take appropriate steps to protect your consulting and development efforts without relying solely upon the password. For more information about accounts and privileges, see Help.

If you have a dispute with your customer, you must resolve this dispute directly with the customer. FileMaker 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 Utilities.

To ensure the quality of your custom database solution:

- Verify every function and option in your solution. If you're developing a solution for both platforms, test it on both Windows and Mac OS X platforms.
- Make sure your runtime database solution does not use a standard FileMaker Pro feature that is hidden or disabled in the runtime application. See "Feature comparison of the runtime application with FileMaker Pro" on page 71.
- Verify that all scripts and buttons work as expected. This is especially important if you're displaying your solution in Kiosk mode. See "About Kiosk mode" on page 39.
- 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 full access privileges, 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 33 and "Your responsibilities as a developer" on page 35.

Important You should keep an unbound version of any runtime database solution files, especially if you've permanently removed full access privileges. See "Removing full access privileges from databases" on page 27.

Distributing updates to runtime database solutions

If you make feature enhancements or modifications to the primary bound file of your runtime database solution, you can distribute the updated file to your users without rebinding it. If you change the filename 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 are distributing a new auxillary file that requires new file references in the main file or that requires other files to interact with it, you must update all files that have been modified.

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 from your copy of the runtime solution in FileMaker Developer.

2. Make the changes to the primary file.

3. If necessary, create an Import script so users can import their existing data into the new primary file.

See "Importing data into upgraded runtime solutions" on page 26.

4. 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 Developer, create the new auxiliary file or open the original auxiliary file (before it was bound) and make changes as required.

2. If necessary, create an Import script so users can import their existing data into the new file.

See "Importing data into upgraded runtime solutions" on page 26.

3. Use the Developer Utilities to rebind all of the files in the runtime database solution and include the new or updated auxiliary file.

Use the same binding key that you used for the primary file. Remember that the binding key is case-sensitive. See "Binding databases into runtime database solutions" on page 22.

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.

Chapter 5
Customizing database solutions

You can use FileMaker Developer to further customize your solutions:

- Use the Developer Utilities to create a solution that displays your database in Kiosk mode. When users without administrative accounts open a Kiosk solution, it displays on a full screen and without toolbars, menus, or other window controls.
- Use ScriptMaker to create scripts that can then be attached to buttons. The runtime application does not have menu commands or toolbar buttons to open or close other files. Scripts attached to buttons must be used in runtime solutions and Kiosk solutions to open or close other files. Scripts and buttons can be used for other controls in database solutions.
- Use the Developer Utilities to customize the About, Help, and Scripts menus in database solutions and runtime database solutions.

About Kiosk mode

Kiosk mode is a way of displaying your database solution or your runtime database solution on a full screen, without any toolbars or menus. As the name suggests, Kiosk mode can be used to present your database to users as an information kiosk. You can design your database to run through a touch screen.

Database files that open in FileMaker Pro and FileMaker Developer, or in the runtime application can both be transformed into files that accounts with specific privilege sets must open in Kiosk mode. When you create a solution to run in Kiosk mode, you need to consider how users will navigate your solution and how they will quit your solution. For more information about accounts and privileges, see Help.

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, FileMaker Developer, or the runtime application.

To decide how users will navigate your Kiosk 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 solution 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 Enter Preview Mode script step is followed by a Pause/Resume Script script step and specify an amount of time the script should remain in Preview mode. Place an Enter Browse Mode script step after the Pause/Resume Script script step.

Note When a Kiosk 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.

Closing Kiosk solutions

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

To ensure that users can access the primary file and quit the application cleanly:

- In each auxiliary file, provide a startup script that opens the primary file.
- In each auxiliary file, place a "Main Menu" button that runs the startup script to open the primary file.
- In the primary file, include an Exit button.

For information about creating buttons and scripts that emulate missing menu options and window controls, see "Emulating menu commands and window controls" on page 41.

Creating Kiosk solutions

Kiosk mode is ignored if the solution is opened by accounts with the Full Access privilege set, a privilege set that allows management of extended privileges, or a privilege set that allows modification of layouts, value lists, and scripts.

For your solution to display in Kiosk mode, you must:

- create an account with a limited privilege set or create a specific Kiosk account.
- enable Kiosk mode. At the same time that you enable Kiosk mode, you can bind the database as a runtime solution.
- clear the default option of logging into the file with the Admin account.

To create a Kiosk account:

1. With the database solution open, choose File menu > Define > Accounts & Privileges.
2. In the Define Accounts & Privileges dialog box, click New.
3. In the Edit Account dialog box, type an account name, click Active for the Account Status, and select New Privilege Set from the Privilege Set list.
4. In the Edit Privilege Set dialog box, give the privilege set a name and description.
5. For Layouts, Value Lists, and Scripts, select either All view only or All no access.
6. Clear the Manage extended privileges checkbox.
7. Select other options as required.
8. Click OK.

To enable Kiosk mode:

1. Follow steps 1 to 9 in "Modifying database solution files" on page 18.
2. In the Specify Solution Options dialog box, select Enable Kiosk mode for non-admin accounts.
3. Select other options as required.
4. Click OK.
5. To be able to quickly repeat the process, click Save Settings, and choose a folder and location for your settings file.

   See "Saving and reusing Developer Utilities settings" on page 25.
6. Click Create.

If you did not bind the files to a runtime application, the Developer Utilities copy the selected database files to the Project Folder. If you did bind the files to a runtime application, the Developer Utilities copy all of the runtime files to a new folder created inside the Project Folder and named after the runtime solution.

To change the default option of logging into the file with the Admin account:

1. With the database solution open, choose File menu > File Options.
2. On the Open/Close tab, clear Log in using.
3. Click OK.

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. For more information about creating scripts to automate tasks, see Help.

Note Use the Debug Scripts option on the Scripts menu to test scripts. See “Using the Script Debugger” on page 12.

Creating startup scripts

Startup scripts are useful for controlling what appears when users open a file in your solution. You can control which layout is displayed and the size, position, and magnification of the window. For a startup script example, see Help.

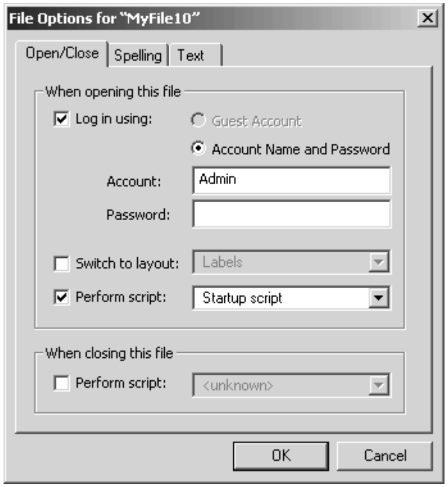
To create a startup script:

1. Open the primary file of the database.

2. Choose Scripts menu > ScriptMaker and create a script to perform the desired actions.

3. Choose File menu > File Options > Open/Close tab.

4. For When opening this file, select Perform script, then from the drop-down list choose the script you created.



File Options for "MyFile10"

When opening this file

☑ Log in using: ○ Guest Account ● Account Name and Password

Account: Admin

Password:

☐ Switch to layout: Labels

☑ Perform script: Startup script

When closing this file

☐ Perform script: <unknown>

An example of a file-specific File Options dialog box

5. Click OK.

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

Centering database windows in Kiosk screens

To center your databases in the middle of a Kiosk screen, create a startup script that uses the Adjust 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 Adjust Window script step, perform any script steps that affect the window display area (such as Go to Layout or Show/Hide Status Area). Once the window area is determined, add the Adjust Window script step.

The Adjust Window script step may cover up a window that has an Exit 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 script steps attached
Menu commands	Script steps for any menu command (for example, Sort Records, Print, and Open Preferences)
Zoom controls	Adjust Window or Set Zoom Level
Status area control	Show/Hide Status Area
Mode pop-up menu	Enter Browse Mode, Enter Find Mode, or Enter Preview Mode

To emulate these interface elements	Create buttons with these script steps attached
Vertical scroll bar	Scroll Window (if the layout is longer than one screen)
Horizontal scroll bar	Scroll Window (if the layout objects are wider than one screen)
Window size and location	Move/Resize Window

Creating dynamic buttons

By using scripts and calculations to evaluate the state of button fields, you can make dynamic buttons that change each time they are clicked.

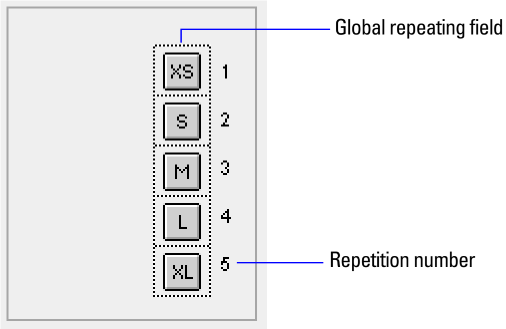
To create dynamic buttons:

1. Define the dynamic button field.
2. Create the dynamic button script.
3. Connect the field and the script.

The following example shows how to create a button that changes every time it is clicked.

To define the dynamic button field:

1. Choose File menu > Define > Database > Fields tab.
2. Create a field named `Icons` and make it a Container type.
3. Click Options > Storage tab.
4. Select the Use global storage checkbox and type the number of button states for the Maximum number of repetitions.
5. Click OK.
6. In Browse mode, select the Icons field and choose Insert menu > Picture to add graphics to the Icons field.



Graphics buttons added to a repeating field

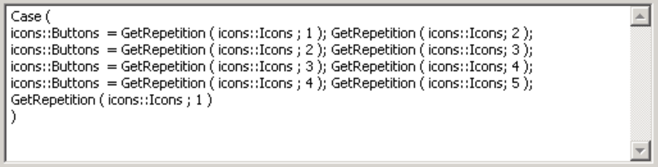
Tip Ensure that all graphics are the same size.

7. Choose File menu > Define > Database > Fields tab.
8. Create a container field named `Buttons`.
9. Click Options > Auto-Enter tab.
10. For Calculated value, click Specify and enter the function

    `GetRepetition (Icons; 1)`
11. Click OK.

To create the dynamic button script:

1. Choose Scripts menu > ScriptMaker and click New.
2. Name the script `Toggle Buttons`.
3. Add a `Set Field` script step.
4. For Specify target field, click Specify.
5. Double-click Buttons.
6. For Calculated result, click Specify.
7. Write a Case function that evaluates each GetRepetition test expression for the icon number and increments the number by one.



```
Case (
icons::Buttons  = GetRepetition ( icons::Icons ; 1 ); GetRepetition ( icons::Icons; 2 );
icons::Buttons  = GetRepetition ( icons::Icons ; 2 ); GetRepetition ( icons::Icons; 3 );
icons::Buttons  = GetRepetition ( icons::Icons ; 3 ); GetRepetition ( icons::Icons; 4 );
icons::Buttons  = GetRepetition ( icons::Icons ; 4 ); GetRepetition ( icons::Icons; 5 );
GetRepetition ( icons::Icons ; 1 )
)
```

Case function for dynamic buttons

8. Click OK.

To connect the field and the script:

1. Choose View menu > Layout Mode.
2. Select the Buttons field and choose Format menu > Field Behavior.
3. Clear both Allow field to be entered checkboxes and click OK.
4. Choose Format menu > Button.
5. Select Perform Script and specify Toggle Buttons.
6. Click OK.

You can switch to Browse mode and test your dynamic button. For more information about using buttons with scripts, see Help.

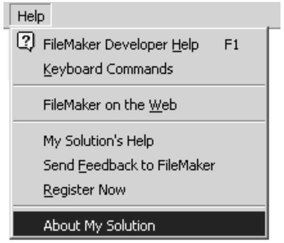
Customizing About, Help, and Scripts menus

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

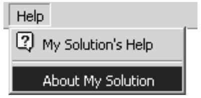
Adding custom scripts to the About menu command

Use the Developer Utilities to add a custom script to the About menu command for your database solution or your runtime solution. The custom script will bring the user to the About layout that describes your solution and provides users necessary information about it.

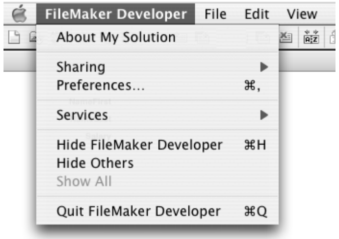
When the Developer Utilities process your database files, they create a menu command named "About <your solution>" and place it in the Help menu (Windows) or Application menu (Mac OS X). For more information, see "Creating an About layout" on page 33.



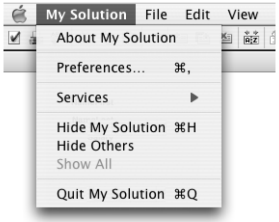
Custom About command in FileMaker Developer (Windows)



Custom About command in a runtime application (Windows)

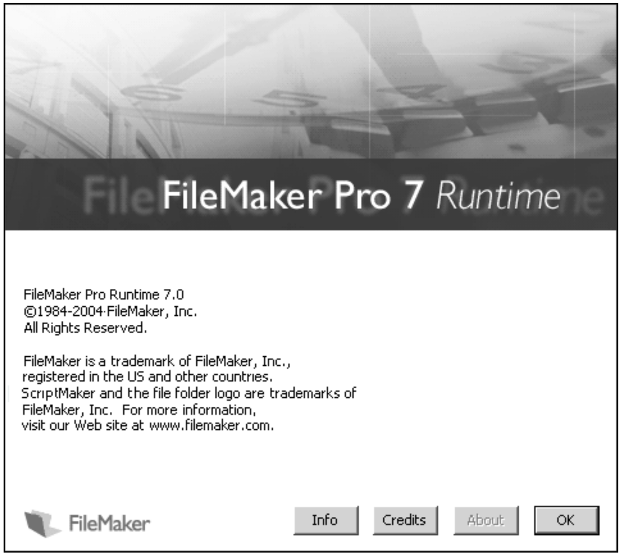


Custom About command in FileMaker Developer (Mac OS X)



Custom About command in a runtime application (Mac OS X)

In runtime applications, the custom About command replaces the About FileMaker Pro Runtime 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 Application menu (Mac OS X) in the runtime application.



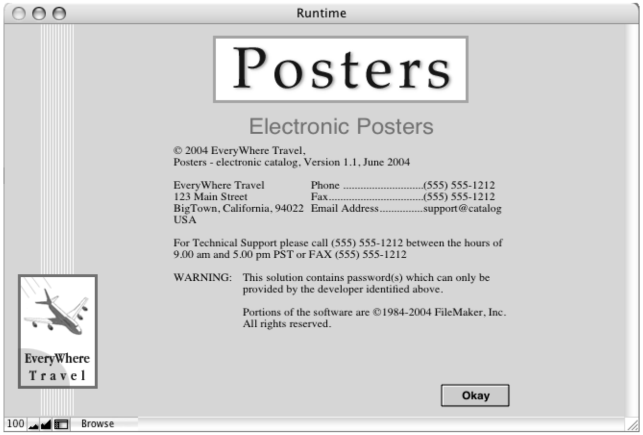
FileMaker Pro 7 Runtime

FileMaker Pro Runtime 7.0
©1984-2004 FileMaker, Inc.
All Rights Reserved.

FileMaker is a trademark of FileMaker, Inc.,
registered in the US and other countries.
ScriptMaker and the file folder logo are trademarks of
FileMaker, Inc. For more information,
visit our Web site at www.filemaker.com.

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

Note In runtime database solutions, you are required to include an About layout 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, FileMaker employees are alerted not to provide technical support to unauthorized users attempting to open the solution. For more information, see "Abiding by the license agreement" on page 9 and "Your responsibilities as a developer" on page 35.



Runtime

Posters

Electronic Posters

© 2004 EveryWhere Travel,
Posters - electronic catalog, Version 1.1, June 2004

EveryWhere Travel
123 Main Street
BigTown, California, 94022
USA

Phone ....(555) 555-1212
Fax..................................(555) 555-1212
Email Address...............support@catalog

For Technical Support please call (555) 555-1212 between the hours of 9.00 am and 5.00 pm PST or FAX (555) 555-1212

WARNING: This solution contains password(s) which can only be provided by the developer identified above.

Portions of the software are ©1984-2004 FileMaker, Inc. All rights reserved.

The About layout in this 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:

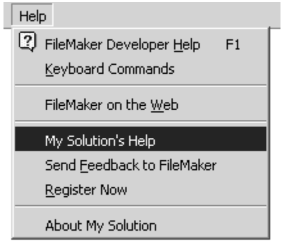
1. Follow steps 1 to 9 in "Modifying database solution files" on page 18.
2. In the Specify Solution Options dialog box, select Custom Script for About menu item.
3. Select a script from the Script name drop-down list.
4. Select other options as required.
5. Click OK.
6. Click Create.

If you did not bind the files to a runtime application, the Developer Utilities copy the selected database files to the Project Folder. If you did bind the files to a runtime application, the Developer Utilities copy all of the runtime files to a new folder, created inside the Project Folder and named after the runtime solution.

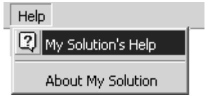
For more information, see "Creating an About layout" on page 33.

Adding custom scripts to the Help menu command

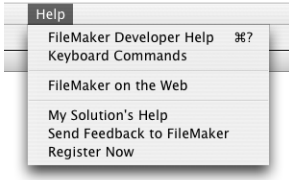
You can create your own help system for your solution. Create a Help layout or file and use a script to open it. With the Developer Utilities, you add the script name to the Help menu. In runtime applications, the custom Help script command replaces the FileMaker Developer Help command. For more information, see "Creating a custom Help layout" on page 34.



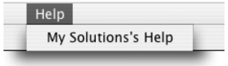
Custom Help command in FileMaker Developer (Windows)



Custom Help command in a runtime application (Windows)



Custom Help command in FileMaker Developer (Mac OS X)



Custom Help command in a runtime application (Mac OS X)

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

1. Follow steps 1 to 9 in "Modifying database solution files" on page 18.

2. In the Specify Solution Options dialog box, select Custom Script for Help menu item.

3. Select a script from the Script name drop-down list.

4. Select other options as required.

5. Click OK.

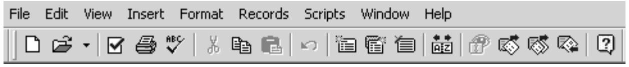
6. Click Create.

If you did not bind the files to a runtime application, the Developer Utilities copy the selected database files to the Project Folder. If you did bind the files to a runtime application, the Developer Utilities copy all of the runtime files to a new folder, created inside the Project Folder and named after the runtime solution.

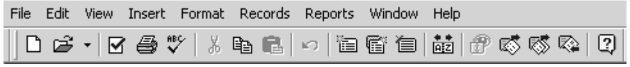
For more information, see "Creating a custom Help layout" on page 34.

Renaming the Scripts menu

Use the Developer Utilities 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 Developer and in a runtime application.



Database solution with Scripts menu



Database solution with renamed Scripts menu

Note The menu name must not exceed 30 characters.

To rename the Scripts menu for your solution:

1. Follow steps 1 to 9 in "Modifying database solution files" on page 18.
2. In the Specify Solution Options dialog box, select Custom Scripts Menu name.
3. Type a menu name.

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

1. Select other options as required.
2. Click OK.
3. Click Create.

If you did not bind the files to a runtime application, the Developer Utilities copy the selected database files to the Project Folder. If you did bind the files to a runtime application, the Developer Utilities copy all of the runtime files to a new folder, created inside the Project Folder and named after the runtime solution.

Chapter 6
Creating custom layout themes

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

A theme is an Extensible Markup Language (XML) document that can be read and edited in a text editor (such as Notepad for Windows or BBEdit for Mac OS X) or XML editor (such as XML SPY or XMetaL). 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.

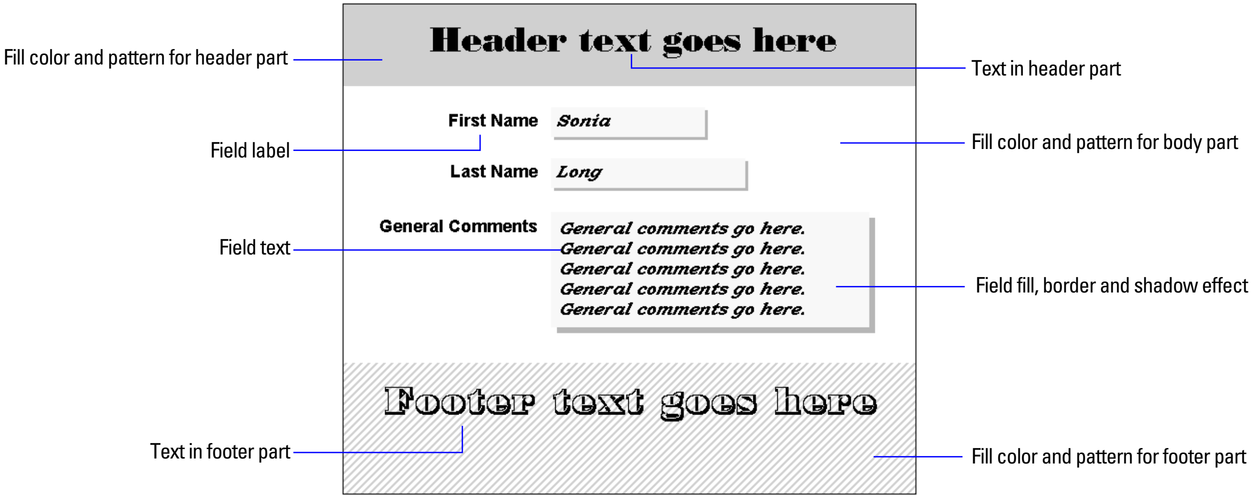
Note A FileMaker theme is not a stylesheet and does not contain positioning information for objects on a layout.

For information about:

- using layout themes and designing layouts, see Help
- XML and its uses, see www.filemaker.com/xml
- publishing your database on the web in XML format, see the PDF manual, *FileMaker Instant Web Publishing Guide*

Modifying FileMaker Developer themes

FileMaker Pro and FileMaker Developer include theme files that can be modified. A 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).


Fill color and pattern for header part
Header text goes here
Text in header part
First Name
Sonia
Field label
Last Name
Long
Fill color and pattern for body part
General Comments
General comments go here.
General comments go here.
General comments go here.
General comments go here.
General comments go here.
Field text
Field fill, border and shadow effect
Footer text goes here
Text in footer part
Fill color and pattern for footer part

Create themes to automatically apply different styles to text and background fills in layout parts, fields, and field labels

Important The XML for a layout theme must be well-formed and comply with the required syntax. See "Requirements for theme files" on page 49 and "Checking theme files for errors" on page 56.

To modify a theme:

1. Make a copy of one of the theme files in the Themes folder.

Windows: FileMaker Developer 7\Extensions\English\Themes\

or

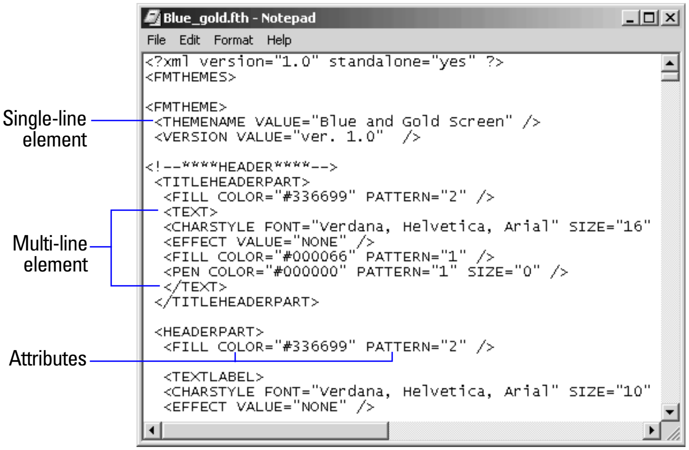
Mac OS X: FileMaker Developer 7/ FileMaker Developer.app/Contents/ Resources/English.lproj/Themes/

Important The total number of theme files is limited to 50.

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 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 a text editor.



```
<?xml version="1.0" standalone="yes" ?>
<FMTHEMES>

<FMTHEME>
 <THEMENAME VALUE="Blue and Gold Screen" />
 <VERSION VALUE="ver. 1.0"  />

<!--****HEADER****-->
 <TITLEHEADERPART>
  <FILL COLOR="#336699" PATTERN="2" />
  <TEXT>
  <CHARSTYLE FONT="Verdana, Helvetica, Arial" SIZE="16"
  <EFFECT VALUE="NONE" />
  <FILL COLOR="#000066" PATTERN="1" />
  <PEN COLOR="#000000" PATTERN="1" SIZE="0" />
  </TEXT>
 </TITLEHEADERPART>

 <HEADERPART>
  <FILL COLOR="#336699" PATTERN="2" />

  <TEXTLABEL>
  <CHARSTYLE FONT="Verdana, Helvetica, Arial" SIZE="10"
  <EFFECT VALUE="NONE" />
```

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 X platforms. For information, see "Valid values for theme attributes" on page 53.

5. Change the values of other elements and attributes.

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" />
```

For guidelines, see the table in "Valid values for theme attributes" on page 53 and "Using values for patterns and colors" on page 55.

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. For information, see "Removing elements from theme files" on page 50.

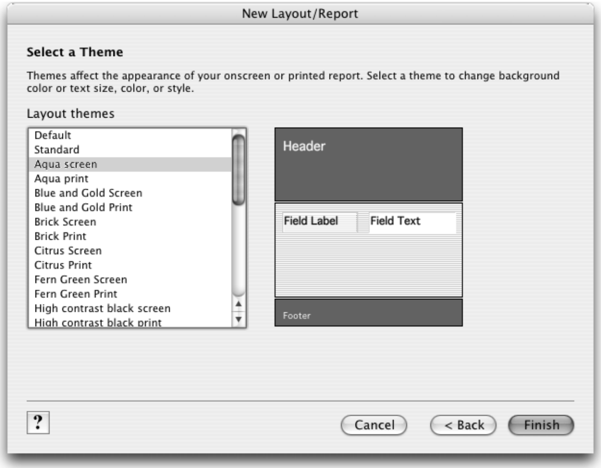
7. Scroll down to the next FMTHEME element and repeat these steps to change the THEMENAME value and other elements.

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

Each new THEMENAME value will appear in the New Layout/ Report assistant as a Layout Theme option.

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

Follow the instruction in the New Layout/Report assistant. The third panel presents you with a list of themes to select from.



Select a Theme

Themes affect the appearance of your onscreen or printed report. Select a theme to change background color or text size, color, or style.

Layout themes

- Default
- Standard
- Aqua screen
- Aqua print
- Blue and Gold Screen
- Blue and Gold Print
- Brick Screen
- Brick Print
- Citrus Screen
- Citrus Print
- Fern Green Screen
- Fern Green Print
- High contrast black screen
- High contrast black print

Names of 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. For information, see "Checking theme files for errors" on page 56.

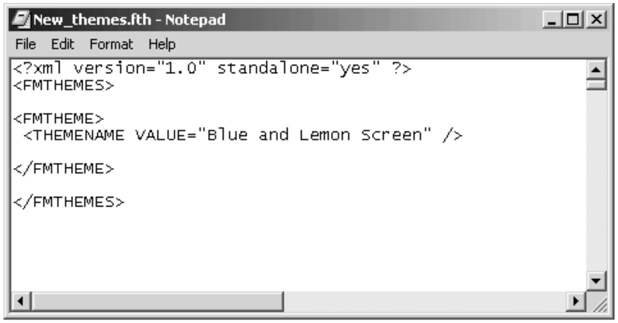
Requirements for theme files

FileMaker Developer 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.

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 Developer will be unable to parse the XML or display the theme in the New Layout/Report assistant.

Minimum 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 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.



```
<?xml version="1.0" standalone="yes" ?>
<FMTHEMES>

<FMTHEME>
 <THEMENAME VALUE="Blue and Lemon Screen" />

</FMTHEME>

</FMTHEMES>
```

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 the New Report/Layout assistant. The order that FMTHEME elements are listed in the file determines the order in which the THEMENAME values appear.

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 (&)	&amp;
less than (<)	&lt;
greater than (>)	&gt;
apostrophe (')	&apos;
quote (")	&quot;

Using the character instead of the character entity results in an error from the XML editor. However, FileMaker Pro and FileMaker Developer do 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 by ignoring 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 X platforms, use the HINT attribute to ensure that upper-ASCII characters (such as the accent mark) appear correctly on both platforms. For more information, see "Valid values for theme attributes" on page 53.

Removing elements from theme files

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, which includes everything inside the element's start and end tags and the start and end tags as well.

FileMaker Developer will use default values for any elements you remove. See "Specifying default values for themes" on page 55.

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

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

A multi-line element has start and end tags on separate lines:

```
<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 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 Developer 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.

This multi-lined element is used	To describe this layout part
<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.
<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 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, such as title text in the header.

XML elements and their theme attributes

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

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.
FMTHEME (required)	VERSION THEMENAME (required) THEMEDEFAULT Any or all layout part elements

These multi-line elements	May contain these elements
TITLEHEADERPART HEADERPART LEADGRANDSUMPART BODYPART TRAILGRANDSUMPART FOOTERPART TITLEFOOTPART	FILL FIELD TEXT TEXTLABEL
LEADSUBSUMPART TRAILSUBSUMPART	FILL FIELD PARTNUMBER TEXT TEXTLABEL
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 *Syntax example:* <CHARSTYLE FONT="**Verdana, Helvetica, Arial**" SIZE="**18**" STYLE="**BOLD, ITALIC**" COLOR="**#FFFFFF**" />
EFFECT	<EFFECT VALUE="**EMBOSS**" />
FILL	COLOR PATTERN *Syntax example:* <FILL COLOR= "**#000066**" PATTERN= "**2**" />
ONOFF	<ONOFF VALUE="**OFF**" />
PARTNUMBER	<PARTNUMBER VALUE="**0**" />
PEN	COLOR PATTERN SIZE *Syntax example:* <PEN COLOR="**#000066**" PATTERN="**2**" SIZE="**0**" />

These single-line elements	Must contain these attributes
SIDES	<SIDES VALUE="**BOTTOM**" />
THEMENAME	HINT (optional) VALUE *Syntax examples:* <THEMENAME VALUE="**Fern Green Print**" /> <THEMENAME HINT="**MAC**" VALUE="**Grün Druck**" />
THEMEDEFAULT	VALUE *Syntax example:* <THEMEDEFAULT VALUE="**CURRENT**"/> For more information, see "Specifying default values for themes" on page 55.
VERSION	VALUE *Syntax example:* <VERSION VALUE="**ver. 1.0**"/> The VERSION element is currently not used by FileMaker Developer, but may be used in future versions.

Valid values for theme attributes

The following table describes the attribute values supported by FileMaker Developer in a layout theme. Values must be enclosed within quotation marks (" "). If a quotation mark is missing, FileMaker Developer 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	RGB color hex values for background fills, text, and borders in layout parts, fields, text blocks, and field labels.
	To display a color, the PATTERN attribute must not be set to "1" (which is transparent).
	FileMaker Developer themes use web-safe palette colors to ensure the color will appear the same on all computers.
	See "Using values for patterns and colors" on page 55.
		COLOR = "#FFFFFF"
		COLOR = "#33FF00"
		COLOR = "#CC9966"
		Or any 6-digit hex value (a combination of numbers 0-9 or letters A-F) preceded by the # symbol.

EFFECT	Embossing, engraving, or drop shadow 3-D effects for a field, text, or field label.
	When used in conjunction with a field border, the line width of the effect will be the same as the border pen size. The pen size value must be greater than zero in order for the effect or border to appear.
		VALUE = "EMBOSS"
		VALUE = "ENGRAVE"
		VALUE = "DROPSHADOW"
		VALUE = "NONE"

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 title case with initial capitals.
		FONT = “Times New Roman”
		FONT = “Geneva”
		FONT = “New York, Times, Helvetica, Arial”
		Or any other available font (In FileMaker Developer, choose Format menu > Font to see the available fonts.)

HINT	The name of the platform that the theme name is edited on and the character set. 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 Developer on both Windows and Mac OS X. Japanese characters are supported if SHIFTJIS is specified.	HINT = “WIN”
		HINT = “MAC”
		HINT = “WIN/ROMAN”
		HINT = “MAC/ ROMAN”
		HINT = “WIN/ SHIFTJIS”
		HINT = “MAC/ SHIFTJIS”

ONOFF	Whether a field’s border should be displayed.	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 Developer 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”

PATTERN	One of 64 valid patterns from the fill pattern palette in FileMaker Developer. Used for background fills in layout parts, fields, text, and field labels, and for borders of fields, field labels, and text.
	See “Using values for patterns and colors” on page 55.
		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 Developer will substitute the closest size.
		SIZE = “36”
		SIZE = “12”
		SIZE = “9”

SIZE
(for the PEN element)
	Thickness in pixels for the outline of text blocks, field labels, and field borders.
	The value for NONE is “0” and the value for HAIRLINE is “-1.”
	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.
		SIZE = “0”
		SIZE = “-1”
		SIZE = “1” through
		SIZE = “8”
		SIZE = “12”

This attribute	Is used to describe these characteristics	And may contain these values

STYLE	Character styles for text in fields, text blocks, and field labels. More than one style can be specified, separated by commas or spaces.
	No error checking is done for contradicting styles, such as UPPERCASE and LOWERCASE.
	The PLAIN style value overrides all other style values.
	STRIKEOUT and STRIKETHRU values are the same.
		STYLE = “PLAIN”
		STYLE = “BOLD”
		STYLE = “ITALIC”
		STYLE = “STRIKEOUT”
		STYLE = “STRIKETHRU”
		STYLE = “SMALLCAPS”
		STYLE = “UNDERLINE”
		STYLE = “WORDUNDERLINE”
		STYLE = “DBLUNDERLINE”
		STYLE = “UPPERCASE”
		STYLE = “LOWERCASE”
		STYLE = “TITLECASE”
		STYLE = “SUPERSCRIPT”
		STYLE = “SUBSCRIPT”
		STYLE = “CONDENSE”
		STYLE = “EXTEND”
		STYLE = “ITALIC, BOLD, SMALLCAPS”

Specifying default values for themes

FileMaker Pro and FileMaker Developer use 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, which are the same values that FileMaker uses when creating a file for the first time.

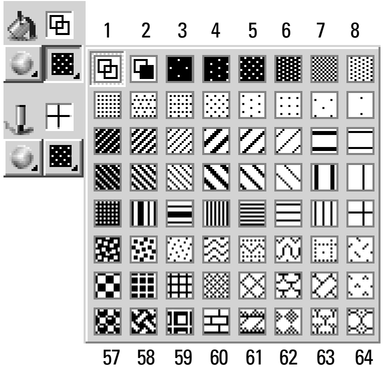
```
<THEMEDEFAULT VALUE=”CURRENT”/>
<THEMEDEFAULT VALUE=”STANDARD”/>
```

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

Using values for patterns and colors

The values for the patterns in the FileMaker Pro and FileMaker Developer pattern palette are numbered consecutively, starting with the top row and counting from left to right, where the value for the pattern in the top left corner is 1. 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 (for example, #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.

The FileMaker Pro and FileMaker Developer color palette contains only web-safe colors. For information on using the color palette, see Help. For more information about using patterns and colors in a layout, see Help.

Adding comments to theme files

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

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

FileMaker Pro and FileMaker Developer 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 and FileMaker Developer.

Checking theme files for errors

FileMaker Pro and FileMaker Developer cannot parse an XML theme document that is not well-formed, and they do not validate the XML in your documents. If one required item is missing or wrong, FileMaker Developer 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>).

Chapter 7
Developing third-party FileMaker plug-ins

If you are a C or C++ programmer and familiar with advanced calculations in FileMaker Pro and FileMaker Developer, you can create external function plug-ins that extend the feature set of the applications. The plug-ins can take advantage of recursion and looping or hook into other programming interfaces. Users can enable your plug-ins in FileMaker Pro, FileMaker Developer, and FileMaker Server and use your external functions in their calculation fields and scripts.

Plug-ins must be registered with FileMaker. The FileMaker 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. For more information, see "Registering your plug-in" on page 68.

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 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 plug-in with FileMaker.
4. Install the compiled plug-in file for your users.

To access your external functions, your users:

1. Enable your plug-in through the Preferences dialog box.
2. Configure your plug-in, if required.
3. Define or edit a calculation field.
4. In the Specify Calculation dialog box, choose Function_Name(parameter 1 ...) as the calculation formula.

To see all external functions, select External functions from the View drop-down list.

About the example plug-in

The example plug-in project is designed to illustrate what a complete 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. You can examine and modify the source code of the example files in any text editor.

The plug-in example includes seven external functions. See "Description of the FMExample plug-in's external functions" on page 61.

The plug-in example files include all the source code required to compile the plug-in for the Windows and Mac OS X platforms. In addition to the plug-in source code, FileMaker Developer includes project files for CodeWarrior Development Studio, Mac OS X Edition 8.3, and Microsoft Visual Studio .NET 2002.

The example plug-in files are located in the English Extras\Examples\FMExample 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 folders and files.

Contents of the FMExample folder

Folder	Description
Example folder	Contains all of the files that are part of the FMExample.
Headers folder	Contains function definition files for the FileMaker API. Do not distribute to users who do not have licenses for FileMaker Developer.
Libraries folder	Contains library files for the FileMaker API. Do not distribute to users who do not have licenses for FileMaker Developer.

Contents of the Example folder

File/Folder	Description
FMPluginExample.cpp	Contains code for implementation of the FMExample.
MacExample.fmplugin	Compiled Mac OS X plug-in package.
MacExample.mcp	CodeWarrior 8 project file.
WinExample.sln	Microsoft Visual Studio .NET project file.
WinExample.vcproj	Microsoft Visual C++ project file, used by WinExample.sln.
WinExample.fmx	Compiled Windows 2000, XP plug-in.
Support folder	Contains all additional resources and code used by FMPluginExample.cpp.

Contents of the Support folder

File/Folder	Description
FMPluginExample.rc	Contains the resources for Windows platform.
FMPluginExample.nib	Contains the resources for Mac OS X platform.
FMPluginExample.strings	Contains the strings for Mac OS X platform.
FMPluginFunctions.cpp	Contains code for implementation of external functions in FMExample.
FMPluginFunctions.h	Contains definitions for external functions, including function IDs.
FMPluginGlobalDefines.h	Contains constants used by the FMExample, including compiler directives to control code compilation.
FMPluginPrefs.cpp	Contains code for implementation of configuration dialog box in FMExample.
FMPluginPrefs.h	Contains definitions for configuration dialog box.
MacExample.plc	Contains bundle definition for Mac platform.
Resource.h	Contains definitions for resource file.

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, FileMaker Developer, or FileMaker Server before they can be used. Some plug-ins must also be configured by the user.

For information on installing Web Publishing plug-ins, see the *FileMaker Server Web Publishing Installation* guide.

To install a plug-in, drag the plug-in file into the Extensions folder inside the FileMaker Developer 7 folder. In Windows, the plug-in extension must be .fmx. In Mac OS X, the plug-in extension must be .fmplugin.

To enable a plug-in:

1. Open the Preferences dialog box.

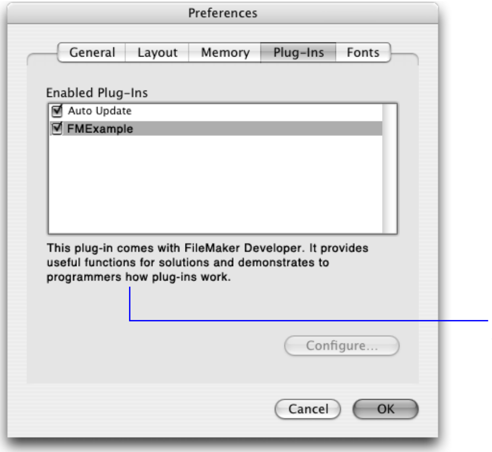
Windows: Choose Edit menu > Preferences.

Mac OS X: Choose FileMaker Developer application menu > Preferences.

2. Click the Plug-Ins tab.

3. Select the plug-in in the list.

A plug-in will appear in the list if it's installed in the correct folder inside the FileMaker Developer application folder.



Enabled Plug-Ins

☑ Auto Update

☑ FMExample

This plug-in comes with FileMaker Developer. It provides useful functions for solutions and demonstrates to programmers how plug-ins work.

Plug-in description text is defined in a resource string

Select a plug-in to enable it

To configure a plug-in:

1. Select the plug-in in the Preferences dialog box.

2. Click Configure.

The Configure button is only available when the sixth character in the option string of the selected plug-in is "Y." See "Option string syntax" on page 65.

3. Follow instructions in the configuration dialog box to configure the plug-in.

4. Click OK.

Description of the FMExample plug-in's external functions

The FMExample plug-in provided in the Microsoft Visual C++ and CodeWarrior example projects adds the following external functions to FileMaker Pro, FileMaker Developer, and FileMaker Server.

Function's name and parameter	Description of external function
XMpl_Add (number1; number2)	Adds number1 and number2 together and returns the result. The function is the same as the plus operator in the calculation engine.
XMpl_Append (textToAppend ...)	Takes a multiple list of parameters, concatenates them, and returns the result. The function is the same as the ampersand operator in the calculation engine.
XMpl_Evaluate (calcToEvaluate)	Takes a simple or complex calculation, evaluates the calculation, and returns the result. Any calculation supported by FileMaker can be passed to this function. The function is identical to the Evaluate function in the calculation engine.
XMpl_NumToWords (number)	Returns a number in bank check format. For example 44.345 returns Forty-Four Dollars and 34 Cents. All digits beyond the second decimal place and any alphabetical characters are ignored.
XMpl_StartScript (filename; scriptname)	Runs the script specified by the scriptname parameter on the file specified by the filename parameter.

Function's name and parameter	Description of external function
XMpl_UserFormatNumber (textOrNumber)	Returns the parameter as a text string formatted as specified in the configuration dialog box. Use this function to format text or numbers such as telephone numbers, postal codes, and so on. Formatting proceeds from right to left. Each # symbol in the format string is replaced by the next character in the parameter string. All remaining # symbols are replaced with zeros. This function demonstrates both client-only functionality and the plug-in configuration dialog.
XMpl_FormatNumber (formatString; textOrNumber)	The same as XMpl_UserFormatNumber, but formatString is provided as a parameter. This function illustrates a function that is visible in the Specify Calculation dialog for Auto Entry, and also supports calls from FileMaker Server and Instant Web Publishing.

Using the example plug-in

To access the external functions:

1. Open the Preferences dialog box.

Windows: Choose Edit menu > Preferences.

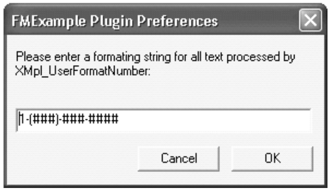
Mac OS X: Choose FileMaker Developer application menu > Preferences.

2. Click the Plug-Ins tab.
3. Select FMExample.

Because the example plug-in includes a function that requires configuration, the Configure button is enabled.

4. Click Configure.

The configuration dialog box that appears depends on how the plug-in source code was written. The XMpl_UserFormatNumber function in the FMExample plug-in displays the following configuration dialog box.



Please enter a formating string for all text processed by XMpl_UserFormatNumber:

1-(###)-###-####

The dialog box that appears when you configure the example plug-in

5. 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.

6. Click OK to close the Preferences dialog box.
7. In FileMaker Developer, choose Define menu > Database > Fields tab.

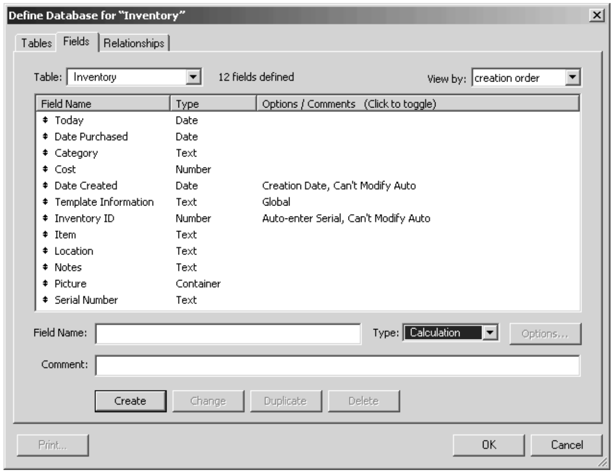
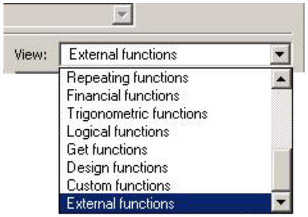


Table: Inventory 12 fields defined View by: creation order

Field Name	Type	Options / Comments (Click to toggle)
Today	Date	
Date Purchased	Date	
Category	Text	
Cost	Number	
Date Created	Date	Creation Date, Can't Modify Auto
Template Information	Text	Global
Inventory ID	Number	Auto-enter Serial, Can't Modify Auto
Item	Text	
Location	Text	
Notes	Text	
Picture	Container	
Serial Number	Text	

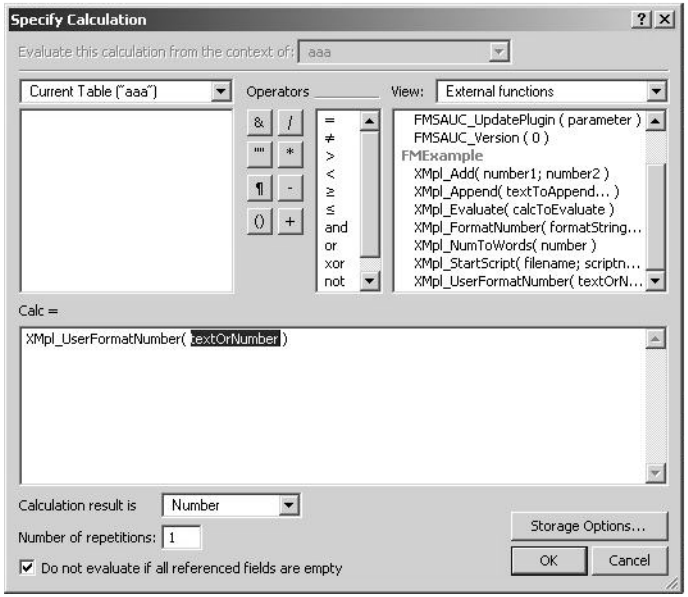
Type: Calculation

8. Create a calculation field.
9. In the Specify Calculation dialog box, choose External Functions from the View drop-down list.


View: External functions

10. Double-click an external function to add it to the formula box.

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.


Specify Calculation

Evaluate this calculation from the context of: aaa

Current Table ("aaa")

View: External functions

Calc = XMpl_UserFormatNumber(textOrNumber)

Calculation result is: Number

Number of repetitions: 1

☑ Do not evaluate if all referenced fields are empty
? ×
Operators
& / "" * ¶ - () +
= ≠ > < ≥ ≤ and or xor not
FMSAUC_UpdatePlugin (parameter)
FMSAUC_Version (0)
FMExample
XMpl_Add(number1; number2)
XMpl_Append(textToAppend...)
XMpl_Evaluate(calcToEvaluate)
XMpl_FormatNumber(formatString...
XMpl_NumToWords(number)
XMpl_StartScript(filename; scriptn...
XMpl_UserFormatNumber(textOrN...
Storage Options...
OK
Cancel

11. Replace the parameter placeholder with the required parameter or parameters for the function.

12. Continue to build the formula then and click OK when you're done.

13. Click OK to close the Define Database 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 items:

- version information in FMPluginExample.strings and FMPluginExample.rc
- plug-in and function names in FMPluginExample.strings and FMPluginExample.rc
- configuration function in FMPluginPrefs.cpp
- external function definitions and coding in FMPluginFunctions.cpp

Customizing the example resources

You must make the following modifications to the plug-in resource files to create a custom external function plug-in:

- Modify the version variables and strings to meet your needs.
- Revise the configuration dialog box to meet your needs.
- Specify the correct option string values.
- Edit plug-in names and description.
- Define your function names and function prototypes.

Customizing FMPluginExample.cpp

Make your modifications to the FMPluginExample.cpp in the functions listed in the following table.

Function name	Customization
Do_PluginInit	Provide your own unique plugin ID for "pluginID." Register each function, providing its name, description, and function to be used. Call fmx::ExprEnv::RegisterExternalFunction to register your functions.
Do_PluginIdle	Add any idle processing your plug-in needs.
Do_PluginShutdown	Revise the UnRegisterExternalFunction calls to reverse the registration done in Do_PlugInit. Call fmx::ExprEnv::UnRegisterExternalFunction to unregister your functions.

Customizing FMPluginPrefs.cpp

This file contains the Do_PluginPrefs function for the implementation of the configuration dialog box. Revise or remove this code as needed.

Customizing FMPluginFunctions.cpp

Revise or remove the functions provided in the FMPluginFunctions.cpp file and define your own. Do_PluginInit refers to these functions when evaluating external functions in calculations.

Requirements for writing external function plug-ins

FileMaker 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 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.

API code files

There are ten API code files in the Headers folder: FMXExtern.h, FMXCalcEngine.h, FMXBinaryData.h, FMXDateTime.h, FMXTextStyle.h FMXTypes.h, FMXFixPt.h, FMXClient.h, FMXText.h, and FMXData.h. The files are not redistributable in source code (or human readable) form, cannot be modified, and are only provided to enable licensees of FileMaker Developer to compile plug-ins for use with FileMaker products. Not all the files are required to build all types of plug-ins.

The FMXExtern.h is absolutely required. The FMXExtern.h defines the parameter block (the shared data structure used by your plug-in and FileMaker Pro, FileMaker Developer, or FileMaker Server) and some shared function calls. The function calls are used to manipulate the parameter and result handles in the parameter block.

The FMXExtern.h file defines the call-back functions for backward compatibility operations and the different kinds of plug-in events (FileMaker Pro, FileMaker Developer, or FileMaker Server messages) sent to the plug-in in a FMExternCallSwitch definition.

FMExternCallStruct defines the structure of the parameter block. FMExternCallPtr is a pointer to that structure and gFMExternCallPtr is a global variable that should be defined in your code.

The FMXCalcEngine.h file contains the register and unregister functions. It will be used in most plug-ins, as the plug-ins will likely need to register functions.

The functionality of the remaining API code files is described in comments that are included in the files themselves.

Option string syntax

The option string must be 11 characters long for plug-ins.

The first four characters of the option string are the ID of the plug-in. The ID must be unique for each plug-in and must not begin with "F," "FM," or "Web." For the Mac OS X, 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 X version of your plug-in. To register plug-in IDs as Creator codes, go to the developer support pages on the Apple Computer web site at www.apple.com/developer. For more information, see "Registering your plug-in" on page 68.

The fifth character of the option string is always "1" and the eight, tenth, and eleventh are always "n." Other values for these flags are reserved for FileMaker use only.

For example, "Moc31YnnYnn" is a option string for a plug-in with the ID of "Moc3" (characters 1-4) that requires configuration (character 6 = "Y"), uses the new style registration and functions callbacks (character 7 = "n"), and requires special idle time (characters 9 = "Y").

Table of option string characters

Characters in the option string	Description of characters
1-4	Characters 1-4 are the plug-in ID. Register the ID as a Creator code on the Apple Developer Support web site at www.apple.com/developer.
5	Character 5 is always "1."
6	Set the sixth character of the option string to "Y" if you want to enable the Configure button for plug-ins in the 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 kFMXT_DoAppPreferences message. For more information, see "FileMaker messages sent to the plug-in" on page 66.
7	Set to "n" for the new style plug-in registration and function callbacks demonstrated in the FMExample. Only set to "Y" if your plug-in requires the legacy function string list and single external callback.
8	Character 8 is always "n."
9	Set the ninth character of the option string to "Y" if the kFMXT_Idle 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	Character 11 is always "n."

Naming conventions for external functions

The function name prefix for all of the plug-in's external functions must be a unique value containing four or five characters and must not begin with the characters "FM" or "Web." Four-character prefixes are reserved by FileMaker. For example, the FMPluginExample plug-in's function name prefix is "XMpl."

FileMaker will manage the naming conventions for plug-in name, filename, and function prefix. For this reason, you need to register your plug-in. FileMaker has reserved certain naming conventions for external functions. For more information, see "Registering your plug-in" on page 68.

FileMaker messages sent to the plug-in

There are six possible calls that FileMaker Pro, FileMaker Developer, or FileMaker Server can request of your plug-in. Messages sent to your plug-in are supplied in the whichCall field of the parameter block, FMExternCallStruct, defined in the FMXExtern.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 Function message received by legacy plug-ins that set character 7 in the options string to "Y" and that register their functions the old external way
- kFMXT_GetString — the GetString message received by plug-ins that use the new style of registration when the plug-ins provide the option string, plug-in name, and description

Initialization message

The Initialization message, kFMXT_Init, is sent to the plug-in whenever it is enabled in FileMaker Pro, FileMaker Developer, or FileMaker Server. This may or may not correspond with the startup of the application, depending on whether the plug-in is enabled in the Preferences dialog box.

There are two 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 API that is incompatible with the API with which it was compiled.
- kCurrentExtnVersion is the only other result value that should be returned. This causes the plug-in to be enabled.

For the FMPluginExample plug-in, 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 that it's compatible with the version of FileMaker Pro, FileMaker Developer, or FileMaker Server 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 Mac OS X, 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.

If the preferences are set properly and the API version is correct, the Do_PluginInit function in the FMPluginExample plug-in will return kCurrentExtnVersion.

After you set the preferences, register each external function by providing its name, description, and the function to be used. Use fmx::ExprEnv::RegisterExternalFunction to register your functions.

Shutdown message

The Shutdown message, kFMXT_Shutdown, is sent to the plug-in whenever it is disabled in FileMaker Pro, FileMaker Developer, or FileMaker Server. This may or may not correspond with the quitting of the application, depending on whether the plug-in is disabled in the Preferences dialog box.

The FMPluginExample 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 operating system services you may be using. It's possible for a plug-in to be enabled and disabled multiple times during a session, so it's important for your plug-in to clean up memory.

Unregister each external function registered during the Initialization message using fmx::ExprEnv::UnRegisterExternalFunction.

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 option string and the plug-in is currently enabled.

There are five times when this message is called by the FileMaker application.

If the idleLevel parameter is not zero, then the routine has been called while the application is running a script or is being controlled by the user. One of the following four messages has been sent::

Message	Meaning
kFMXT_UserNotIdle = 1	The user has done something within the last 30 seconds.
kFMXT_ScriptPaused = 2	The user is running a script that has been paused.
kFMXT_ScriptRunning = 3	The user is running a script.
kFMXT_Unsafe = 4	Same as if the unsafeCalls parameter is set to true.

Do not perform any lengthy, user interface, or event processing when the idleLevel parameter is not zero.

The Idle message will also be sent is when the application detects free time and does its own internal idle handling.

Message	Meaning
kFMXT_UserIdle = 0	The user hasn't done anything within the last 30 seconds or more.

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 Preferences dialog box.

The plug-in should display a dialog box 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 user interface here. The Configure button will only be enabled if the sixth character of the option string is set to "Y." For more information, see "Option string syntax" on page 65.

Any options that need to be saved should be placed in their own registry entry (Windows) or in their own preference file (Mac OS X).

The FMExample plug-in needs to implement a configuration dialog box for the XMpl_UserFormatNumber function, so the flag has been set in the option string (Xmpl1Ynnnnn) and the function Do_PluginPrefs is called when the Preferences message is received.

External Function message

The External Function message, kFMXT_External, is a legacy message for old style plug-ins. It is no longer required for plug-ins that are registered in the new style.

GetString message

The GetString message, kFMXT_GetString, is sent to the plug-in when FileMaker Pro, FileMaker Developer, or FileMaker Server want to retrieve one of the following strings from the plug-in. The plug-in developer can decide where to store the strings.

String	Meaning
kFMXT_OptionsStr = 131	The option string
kFMXT_NameStr = 128	The plug-in name
kFMXT_AppConfigStr = 129	The help text to display in the Preferences dialog box

Avoiding potential Mac OS X resource conflicts

Problems can occur on Mac OS X machines if your plug-in has the same ID for a resource that FileMaker Pro, FileMaker Developer, FileMaker Server, 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, FileMaker Developer, or FileMaker Server resources. FileMaker 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 X 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, FileMaker Developer, or FileMaker Server 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. For more information, see "API code files" on page 64.

Providing documentation for your plug-in

Your 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 at www.filemaker.com.

Registering your plug-in

Register your external function plug-in with FileMaker 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 plug-in to suit their needs.

Before registering your plug-in, you can search to see if the plug-in name or option string ID you are requesting has already been assigned.

You must register each plug-in separately. To register your plug-in, go to the Support section of www.filemaker.com.

Revising a registered plug-in

If you need to revise information about a plug-in that is already registered to you, you must send an email message to FileMaker 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 plug-in 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 with FileMaker Pro

When you double-click the FileMaker Pro application icon to start the application, the New Database dialog box opens and you can choose 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 Database, Define Value Lists, Define File References, and Define Accounts & Privileges dialog boxes, Layout mode, and ScriptMaker.

- Custom functions created with FileMaker Developer will work in the runtime application, although users of the runtime application cannot modify them or create new custom functions.
- 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 command on a runtime database window.)

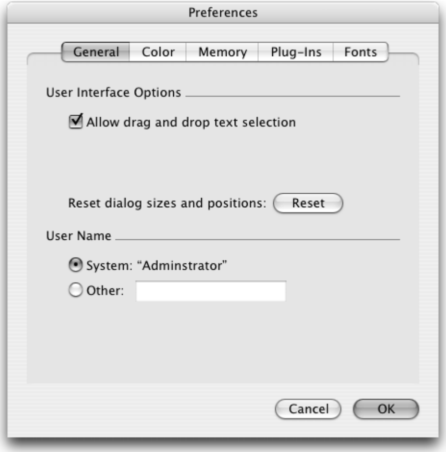
- The Scripts menu can be named something different in the runtime application.
- FileMaker Pro Help is not available in the runtime application. However, the Help menu and the Runtime menu can contain custom Help and About menu commands.
- Some tools are not available on the toolbars in Browse mode, Find mode, and Preview mode in the runtime application.
- External function plug-ins can be enabled in the Preferences dialog box.
- FileMaker Pro File Sharing, serving a database on the web, or communicating with a Java applet requires FileMaker Pro or FileMaker Developer. You can, however, use a compatible version of FileMaker Server to serve runtime solution files.
- 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 are not available on the General tab of the Preferences dialog box:

- Show templates in New Database dialog checkbox
- Show recently opened files checkbox

The Layout tab is changed to the Color tab in the Preferences dialog box for the runtime application.



Preferences

General | Color | Memory | Plug-Ins | Fonts

User Interface Options

☑ Allow drag and drop text selection

Reset dialog sizes and positions: Reset

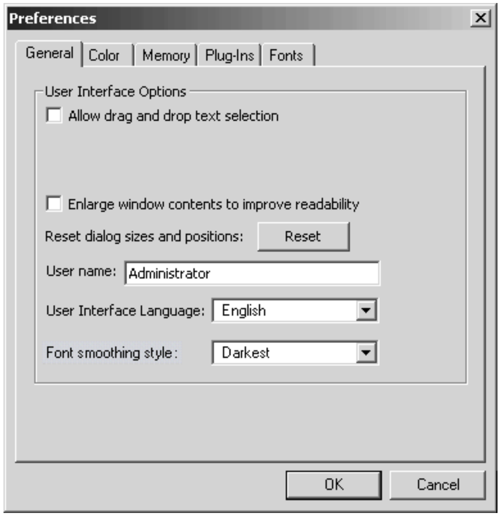
User Name

● System: "Adminstrator"

○ Other:

Cancel | OK

General preferences in a runtime application (Mac OS X)



Preferences

General | Color | Memory | Plug-Ins | Fonts

User Interface Options

□ Allow drag and drop text selection

□ Enlarge window contents to improve readability

Reset dialog sizes and positions: Reset

User name: Administrator

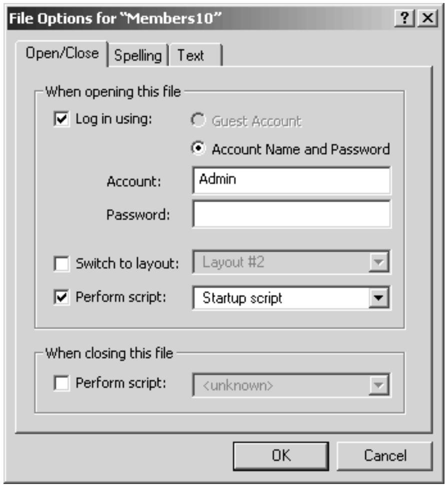
User Interface Language: English

Font smoothing style: Darkest

OK | Cancel

General preferences in a runtime application (Windows)

The File Options dialog box does not have the Open/Close and Text tabs in the runtime application, only the Spelling tab—as shown below.



File Options for "Members10"

Open/Close | Spelling | Text

When opening this file

☑ Log in using: ○ Guest Account ● Account Name and Password

Account: Admin

Password:

□ Switch to layout: Layout #2

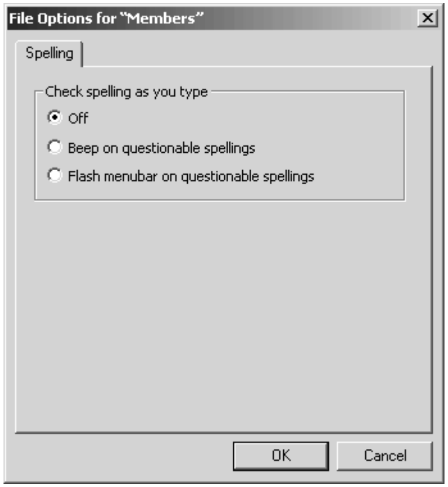
☑ Perform script: Startup script

When closing this file

□ Perform script: <unknown>

OK | Cancel

File Options dialog box in FileMaker Pro



File Options for "Members"

Spelling

Check spelling as you type

● Off

○ Beep on questionable spellings

○ Flash menubar on questionable spellings

OK | Cancel

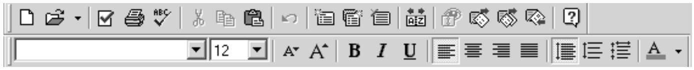
File options dialog box in a runtime application

Toolbar comparison

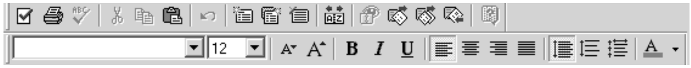
The New Database and Open tools in the standard toolbar (in Browse mode, Find mode, and Preview mode) are not available in the runtime application.

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.



Toolbars in FileMaker Pro



Toolbars in a runtime application

Menu command comparison

The following tables shows the menu commands that are available in FileMaker Pro (Pro) and in the runtime application (RT).

	Windows		Mac OS X	
File Menu command	**Pro**	**RT**	**Pro**	**RT**
New Database	■		■	
Open	■		■	
Open Remote	■		■	
Open Recent	■		■	
Close	■		■	
Define	■		■	
File Options	■	■	■	■
Change Password	■	■	■	■
Print Setup	■	■		
Page Setup			■	■
Print	■	■	■	■
Import Records	■	■	■	■
Export Records	■	■	■	■
Save a Copy As	■	■	■	■
Recover	■	[1]	■	[2]
Exit	■	■		

[1] Press Ctrl+Shift

[2] Press Option+⌘

	Windows		Mac OS X	
Edit Menu command	**Pro**	**RT**	**Pro**	**RT**
Undo	■	■	■	■
Cut	■	■	■	■
Copy	■	■	■	■
Paste	■	■	■	■
Paste Special	■	■		
Clear	■	■	■	■
Select All	■	■	■	■
Find/Replace	■	■	■	■
Spelling	■	■	■	■
Object	■	■		
Export Field Contents	■	■	■	■
Sharing	■			
Preferences	■	■		

	Windows		Mac OS X	
View Menu command	**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	■	■	■	■

	Windows		Mac OS X	
Insert Menu command	**Pro**	**RT**	**Pro**	**RT**
Picture	■	■	■	■
QuickTime	■	■	■	■
Sound	■	■	■	■
File	■	■	■	■
Object	■	■		
Current Date	■	■	■	■
Current Time	■	■	■	■
Current User Name	■	■	■	■
From Index	■	■	■	■
From Last Visited Record	■	■	■	■

	Windows		Mac OS X	
Format Menu command	**Pro**	**RT**	**Pro**	**RT**
Font	■	■	■	■
Size	■	■	■	■
Style	■	■	■	■
Align Text	■	■	■	■
Line Spacing	■	■	■	■
Text Color	■	■	■	■
Text	■	■	■	■

	Windows		Mac OS X	
Records Menu command	**Pro**	**RT**	**Pro**	**RT**
New Record	■	■	■	■
Duplicate Record	■	■	■	■
Delete Record	■	■	■	■
Delete All Records	■	■	■	■
Show All Records	■	■	■	■
Show Omitted Only	■	■	■	■
Omit Record	■	■	■	■
Omit Multiple	■	■	■	■
Modify Last Find	■	■	■	■
Sort Records	■	■	■	■
Unsort	■	■	■	■
Replace Field Contents	■	■	■	■
Relookup Field Contents	■	■	■	■
Revert Record	■	■	■	■

Requests Menu command	**Windows**		**Mac OS X**	
(Find mode)	**Pro**	**RT**	**Pro**	**RT**
Add New Request	■	■	■	■
Duplicate Request	■	■	■	■
Delete Request	■	■	■	■
Show All Records	■	■	■	■
Perform Find	■	■	■	■
Constrain Found Set	■	■	■	■
Extend Found Set	■	■	■	■
Revert Request	■	■	■	■

	Windows		**Mac OS X**	
Scripts Menu command	**Pro**	**RT**	**Pro**	**RT**
ScriptMaker	■		■	
<Script names>	■	■	■	■

	Windows		**Mac OS X**	
Window Menu command	**Pro**	**RT**	**Pro**	**RT**
New Window	■	■	■	■
Show Window	■	■	■	■
Hide Window	■	■	■	■
Minimize Window	■	■	■	■
Tile Horizontally	■	■	■	■
Tile Vertically	■	■	■	■
Cascade Windows	■	■	■	■
Arrange Icons	■	■		
Bring all to front			■	■
<Names of open files>	■	■	■	■

	Windows		**Mac OS X**	
Help Menu command	**Pro**	**RT**	**Pro**	**RT**
FileMaker Pro Help	■		■	
Keyboard Commands	■		■	
FileMaker on the Web	■		■	
Send Feedback to FileMaker	■		■	
Register Now	■		■	
About FileMaker Pro	■		[1]	
About FileMaker Pro Runtime (Displays if no custom About script is specified)		■		[1]
About <runtime solution> (Displays if custom About script is specified)		■		[1]
<Runtime solution Help script name> (Displays if custom Help script is specified)		■		■

[1] See Application Menu command table

	Mac OS X (only)	
Application Menu command	**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)		■
Sharing	■	
Preferences	■	■
Services	■	■
Hide FileMaker Pro	■	

	Mac OS X (only)	
Application Menu command	**Pro**	**RT**
Hide <runtime solution>		■
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:

- Open Define Database
- Open Define Value List
- Open Define File References
- Open Sharing
- Open ScriptMaker
- Open Help (executes custom Help script specified during binding)
- Set Multi-User
- New File
- Open File Options (partially available; Spell checking tab will open)
- Open Remote
- Execute SQL
- 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 or preferences

Windows registry settings

FileMaker Pro stores its registry settings at

HKEY_CURRENT_USER\Software\FileMaker\FileMaker Pro\7.0

FileMaker Developer stores its registry settings at

HKEY_CURRENT_USER\Software\FileMaker\FileMaker Pro\7.0D

The runtime application stores its registry settings at

HKEY_CURRENT_USER\Software\FileMaker\<solution name>\7.0

Note The filename extension for the runtime database files is registered at HKEY_CLASSES_ROOT

Mac OS X preferences

FileMaker Pro stores its preferences in the FileMaker Pro 7.0 Prefs file inside the FileMaker Preferences folder.

FileMaker Developer stores its preferences in the FileMaker Pro 7.0D Prefs file inside the FileMaker Preferences folder. The runtime application stores its preferences in the <Solution name> 7 Prefs file inside the FileMaker Preferences folder.

Index