



FileMaker Go 1.1

Technical Brief

FileMaker Go 1.1

Technical Brief

Version Updates

Version 1.1.1	4
Version 1.1.2	4

FileMaker on the Go

OS Requirements (minimum requirements)

Hosts (supported):	5
Network:	5

Noteworthy Supported Features

Entering Data & Committing Record Changes	6
Text	6
Numbers	6
Dates & Times	7
Field Objects	7
Layout Objects	7
Container Fields	8
Find Mode	8
Sorting	8
Security	8
Scripts	9
Script Triggers	9
Calculations	10
Custom Menus	10
Importing Records	11
Export Field Contents	11
Save Records as PDF	12
Save a Copy As...	12
URL Schemes	12
Script Step, Recognition of Runtime Files and Other Subtle Changes in v1.1	12
FileMaker Server	13

What's Out

Schema Modification	13
Layout Mode & Structure Modification	13

Printing	13
Charting.....	13
Exporting.....	14
Plug-ins	14
Cocoa.....	14
Hosting/Sharing	14
Save/Send Records As.....	14

Performance Considerations

Performance Recommendations	14
Tailoring an Interface for FileMaker Go.....	15
Iconography	15
Log Out Buttons.....	16
Refresh / Request-based Processing	17
Sizing Layout Elements	18
Object Autosizing & Portrait/Landscape Rotations.....	19
Screen Dimensions.....	19
Window Management	20
Zoom Management	20

Scripting for a Mobile World

Scripting Strategies	21
Testing for Platform	21
OnTimer Scripts	21
Using Location Information.....	22
Scripts That Run On Open and Close of a FileMaker Database	22
Related Record Locking Transactions	22
Server Side Queuing	22
Script Logging / Flagging	23

Appendix A: Unsupported Script Steps

Returns Error code 4	23
Returns Error code 3	24
Knowledge Base Articles.....	25
Credits.....	26

FileMaker Go 1.1 Technical Brief

Version Updates

Version 1.1.1

- Create and email a PDF through a scripted process or via menu command
- Save a copy of the database through a scripted process or via menu command
- Export field contents through a scripted process or via menu command
- Container fields can now store photos from the device's image library or images taken by the device camera.
- Pass data from external applications into FileMaker Go using a new URL scheme
- FileMaker Go can now recognize and open files with the ".USR" extension
- Minor script and hibernation changes. Please refer to the "Noteworthy Supported Features" section below

Version 1.1.2

- Application language support for French, Italian, German, Spanish and Japanese
- Save files to a temporary location and overwrite using scripts
- Compatibility issues with large images and container fields have been addressed
- Allows users to restore solutions from hibernation by entering the same account name and password once for multi-file solutions
- General compatibility issues have been addressed

FileMaker on the Go

FileMaker Go® is a full-featured FileMaker® client for iOS. It runs the FileMaker .fp7 file format as either a client to a FileMaker host (Pro or Server) or as local files stored on the device and has much of the same functionality as FileMaker Pro.

The FileMaker Go user experience with FileMaker solutions is analogous to the iOS Safari web browser with full-sized web content. iOS gestures, like pinch-zooming and swipe-scrolling, allow users to comfortably interact with a FileMaker Pro solution that was designed for larger screens. With some caveats, which will be covered in this technical brief, existing FileMaker Pro solutions just work the same as they do in FileMaker Pro. Without any additional development work, your users can log in and access their existing solutions via their iPad and iPhone. FileMaker Go gives FileMaker Pro users the opportunity to create personalized custom mobile solutions without having to hire a programmer to develop applications using iPhone-native Objective C or mobile web apps using tools such as Java or PHP.

That said, while most things work, there are some limitations; please pay special attention to the section on Scripts. Depending on the application and how frequently it is used, some developers will decide to tailor specific layouts and database files for the mobile devices. These mobile layouts could be aimed at smaller screen sizes, or simply provide larger targets for touch. In cases where FileMaker Go is connecting to existing hosted FileMaker solutions, it could be useful to have mobile interfaces for employees like salespeople in the field, allowing them to quickly and easily check the status of an order, enter expenses, and so on, without requiring them to sync or re-enter data later.

FileMaker Go allows users to work with local files stored on their devices. Local access is great for quick reference information or anything that does not need to be shared with other users. For instance, if you wanted to take your existing contact database with you on a trip where web access will be unreliable, you could copy it to your device for use during your trip. Loading and opening files within FileMaker Go is well covered in the product documentation:

For iPhone: <http://www.filemaker.com/products/filemaker-go/for-iphone/help/1/index.html>

For iPad: <http://www.filemaker.com/products/filemaker-go/for-ipad/help/1/index.html>

This technical brief details the FileMaker Pro features that are supported in FileMaker Go, those that are not, and closes with a discussion of best practices. One of the biggest issues to keep in mind is that FileMaker developers must consider client compatibility during testing and development efforts. Additional attention should be paid to all your hosted solutions since you can now expect users to work both via FileMaker Pro on desktops and FileMaker Go on mobile devices, whether or not you designed them with that in mind.

If you are new to FileMaker Go, the FileMaker Go Development Guide is recommended reading and can be found at: http://www.filemaker.com/products/filemaker-go/docs/fmgo_development.pdf

As a companion to this Technical Brief, the screenshots and user interface elements (art, buttons, layout designs) used are all taken from the "FileMaker Go Toolkit" database created by Soliant Consulting. It is available as a free download at <http://www.soliantconsulting.com/gotoolkit/> and contains a collection of assets and examples for developing FileMaker Go user interfaces.

OS Requirements (minimum requirements)

FileMaker Go is a client that accesses either local files stored on your mobile device or databases hosted on FileMaker Server or FileMaker Pro.

- iPhone & iPod Touch: iOS 4.0
- iPad: iOS 3.2 or greater

Hosts (supported):

- FileMaker Server / Server Advanced 10 and 11
- FileMaker Pro / Pro Advanced 10 and 11

Network:

WiFi and 3G networks will work; Edge networks are not supported.

FileMaker Go cannot discover FileMaker Server hosts available through LDAP or view and use SSL certificates from LDAP hosts.

Noteworthy Supported Features

People familiar with using and developing for FileMaker Pro will immediately be struck by the fact that FileMaker Go is FileMaker Pro. There is wide support for most FileMaker Pro features, and to list them all would be both tedious and redundant. FileMaker Go is a full-featured client that works much the same way FileMaker Pro does, minus the development features. There is no Layout mode or other database development tools. You can work with databases, but not create them or modify their schema. Likewise some users may be familiar with Bento® and, from years past, FileMaker Mobile. Both of those products rely on a limited synchronization model for sharing data; FileMaker Go instead delivers a live connection to FileMaker Server and its hosted databases or the option to copy complete database files to and from the iOS device.

This section highlights important FileMaker platform features and explores how they differ on FileMaker Go. If you need an orientation to the interface in FileMaker Go, please refer to its documentation at:

For iPhone: <http://www.filemaker.com/products/filemaker-go/for-iphone/help/1/index.html>

For iPad: <http://www.filemaker.com/products/filemaker-go/for-ipad/help/1/index.html>

Entering Data & Committing Record Changes

As with FileMaker Pro, users can enter a field in Browse mode, make changes, and commit those changes to a database opened either via a network connection or local to the device. When a record is committed, record-level script triggers, auto-entry functions, field validation, and index updates all occur as they do in FileMaker Pro. Tapping out of a field in FileMaker Go is exactly the same as clicking out of a field in FileMaker Pro.

Text

Entering text in a field is fully supported, but there are a few differences when compared with the experience in FileMaker Pro.

When selecting text in FileMaker Go—by double-tapping on text when the keyboard is present—users can cut, copy, paste, and apply spelling corrections using iOS text controls. Note that this mechanism is analogous to the Edit menu commands and spellcheck features in FileMaker Pro.

Users will only be able to edit approximately 64,000 characters of text. In the unusual case of a text field containing more than 64,000 characters, the iOS text control tries to gracefully truncate at a word break as users approach the limit. An error message appears if users try to edit or select text in a field that exceeds 64,000 characters. The text is properly displayed and readable in FileMaker Go, but editing beyond 64,000 characters can only be done in FileMaker Pro.

FileMaker Go properly displays rich text stored in your FileMaker database. However, the following formats are not supported by iOS: highlight, strike through, condense, extend, small caps, superscript, and subscript. These formats will not be displayed but remain stored and displayed in FileMaker Pro.

Bold, italic, underline, title case, upper case, lower case, word underline, and double underline (appears as underline) are all displayed. Manipulation of rich text attributes, creating mixed paragraph styles within a single block of text, entering tab stops, and setting line spacing are not supported. If a user enters a field with rich text attributes and makes any change, FileMaker Go will replace all the unsupported styles with the default attributes of the field. This means users will lose any style and paragraph elements they may have applied to the text in FileMaker Pro.

Auto-complete is not supported on FileMaker Go. Text fields with option “auto-complete using existing values” will behave as though the setting were turned off. Likewise, when entering data, users cannot insert text from an index in FileMaker Go.

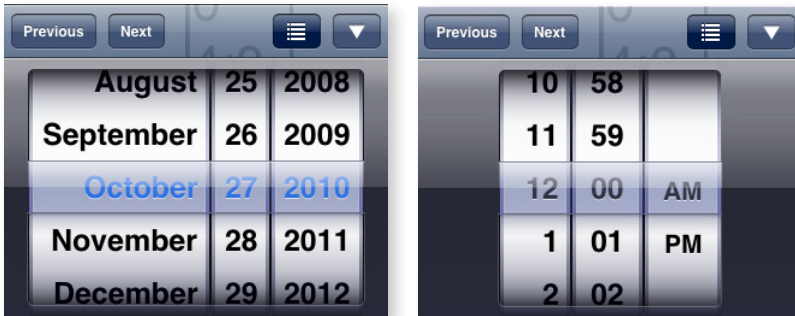
Asian languages are supported although Japanese specific features such as Sideways Text Orientation and Furigana are not supported. IME input will also not work in fields setup with `OnObjectKeystroke` and `OnLayoutKeystroke` script triggers.

Numbers

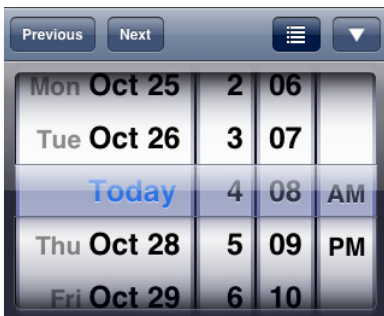
FileMaker Go does not display a “?” character for number fields containing more data than can be displayed as FileMaker Pro does. Instead, it uses the iOS truncation mechanism. The result will also display somewhat differently depending on how alignment is set. If centered or right aligned, you will see the first character followed by an ellipse. For example, the value “8050” may appear as “8...”. If left aligned, the first several characters will be displayed and may not include an ellipse. Using the same example from above, the value “8050” might appear as “805” when left aligned. As a result, consideration should be given to these behaviors and how they will affect users when placing number fields on a layout.

Dates & Times

Dates and Times work just as they do in FileMaker Pro; however, they use iOS controls for data entry as shown below.



The Timestamp iOS control, as seen below, does not support inserting seconds (or fractional seconds); however, you can type seconds into any Time or Timestamp field before or after using an iOS control.



There is no direct means of inserting the Current Date, Time, or Timestamp via keyboard shortcuts or menus. If you wish to provide such functionality, you will need to write scripts for those behaviors and optionally associate those scripts with buttons placed on your layouts. Additionally, script triggers such as the `OnObjectEnter` script trigger could be used to enter the desired data as a starting point for data entry.

Field Objects

Field controls including repetitions, check boxes, radio buttons, and iOS-style value pickers are all supported.

Fields with applied data formatting and conditional formatting display the same in FileMaker Go as in FileMaker Pro.

To remove or clear values from radio buttons in FileMaker Go, you will need to provide a button and/or scripted method.

Layout Objects

Tab Controls, Buttons, Portals, and Web Viewer objects are all supported. Layouts on the iPad/iPhone will look nearly identical to how they appear in FileMaker Pro. Most object formatting options such as pen color, fill color, line width, and embossing/engraving/shadows are supported, as are geometric objects including lines, rectangles, rounded rectangles, and circles. Fill patterns are the exception - FileMaker Go ignores them.

Tooltips are not supported and do not appear in FileMaker Go. In a touch interface there is no hover or rollover state to invoke tooltips.

Container Fields

Container fields support all image formats supported by iOS: GIF, PNG, JPG, and TIFF. Other file formats can be stored (including PDF, Word, and Excel), but they will neither be able to be opened nor have their contents displayed as a thumbnail. Sound and video formats are also not supported; they can be stored, will show as files with file names, but will not execute when you tap on them. Tapping an empty Container field lets users insert a photo taken from the camera or stored in the device's image library. If the Container field has data, users may also choose to Open or Email the contents using the device's local email client. You can cut, copy, paste, and delete by first tapping on a container field, which will expose the iOS keyboard, and then tapping again to access the contextual menu containing these menu items. If you have previously copied an image to the iOS clipboard, you are able to paste it into a container field from this menu. You can also paste from between containers or from a container field to Mail, for example.

Find Mode

Familiar FileMaker Pro operations are supported with both Find Mode and Quick Find, including multiple find requests and constrain/extend options. Although there is no list of operators available in Find Mode, users may enter them manually.

Sorting

Sorting, including custom sort order and ascending/descending operations, is supported. You can only sort by fields on the current layout, however hidden fields can be used to offer more options in the Edit Sort Order window.

In Table View, users can sort by tapping on column headers. The first tap will sort ascending, the second will sort descending and the third tap unsorts the records.

Security

Both file-access security and access to features and operations within a FileMaker Pro database remain unchanged. FileMaker Go does not provide a way to enter a direct file path, so opening files set to not display in the Open Remote File dialog (hidden files) is not supported. In order to work with a hidden file, a developer must create an "opener file" that opens the hidden file through a script.

FileMaker Go does introduce a new "fmrestorelogin" extended privilege. To take advantage of this extended privilege, you must create a new extended privilege using "fmrestorelogin" (no quotes) as the specific keyword.

FileMaker Go, by default, will not save the user name and password of a file that is terminated. The next time FileMaker Go is launched, it will prompt the user for their credentials and attempt to restore the user's prior session, except in the following cases:

- The current user is logged in using a Guest account. FileMaker Go will let the user immediately reconnect as a Guest.
- The current user is logged into a FileMaker database where access to the file was defined in the File Options.
- The current user is logged in using an account that is associated with the "fmrestorelogin" extended privilege. This privilege exists to allow users to resume an interrupted FileMaker Go session without being further stalled by an authentication dialog. In these cases it may be prudent to recommend that all users lock their devices using the iOS security code feature.

The "fmrestorelogin" extended privilege works with **Auto-Restore Login** application setting (Home screen > Settings Menu) in FileMaker Go. If the setting is turned off, a user will always be prompted for their credentials.

If FileMaker Go is closed as a result of a user pressing the Home button or an incoming call, multi-file solutions will no longer prompt the user for credentials for every file when "awoken". FileMaker Go v1.1.2 simply prompts a user for credentials when opening the first file. As long as the other files in the solution contain the same account and password, no additional credentials prompt will be displayed.

The FileMaker Go client should honor all of a solution's existing security settings. This means anyone authorized to access a database via FileMaker Pro is similarly authorized to access that same database, with all the same privileges, in FileMaker Go.

Scripts

Scripts are available to run in FileMaker Go just as they are in FileMaker Pro. As with Web Publishing, some script steps are not supported and some discussion on scripting is warranted; please refer to the "What's Out" and "Scripting for a Mobile World" sections below. The ability to edit scripts is not available in FileMaker Go, nor is there a Script Debugger.

One behavioral change is worth noting from the start. When a script is running in FileMaker Go and is set to allow user abort, any tap on the screen will bring up an abort dialog. In FileMaker Pro the only way a user can abort a script is by pressing the escape key or Command + period (Mac). Given that tapping the screen is a fairly common practice for users—especially if they may not realize a script has not finished running—developers may want to disallow user aborts (by using the **Allow User Abort [Off]** script step) more frequently.

Another important reality of working in a mobile world is that scripts can be aborted at any time, regardless of the **Allow User Abort** settings and ordinary operations of FileMaker. When a user either switches to the Home screen (by pressing the Home button) or receives an incoming phone call on the iPhone (which closes FileMaker Go), all scripts running will be halted without warning.

A third way scripts may not run as expected is when FileMaker Go encounters an unsupported script step. There are two ways in which FileMaker Go handles this. Some script steps are fairly benign and are simply skipped or ignored and the script will run to its conclusion. If you use **Get(LastError)=3** you can check for these cases.

For script steps that could potentially harm your database (for example, Export Records), FileMaker Go will present an alert to the user and, if **Allow Abort** is set to **On**, the user can choose to continue the script or halt its execution. If **Allow Abort** is set to **Off**, FileMaker Go will continue the script after ignoring the problematic script step. It is important that developers test their databases and review their Database Design Reports for script steps that will not run on FileMaker Go.

There is more discussion later in this technical brief on how to approach scripting for FileMaker Go.

Some behavioral differences to note for scripts include (excerpted from the Developer's Guide):

- **Send Mail** does not support Perform without dialog. The email displays on the device, and you can send it manually. This does not apply to mail sent through SMTP.
- A script using **Replace Field Contents** cannot prompt users in FileMaker Go to specify the field and data to be replaced. A **Replace Field Contents** script step with **Perform** without dialog deselected will not be performed.
- The **Select All** and **Set Selection** script steps do not highlight text if the keyboard is hidden on the device.
- The **Open URL** script step using a Google maps URL will open a browser, and then choosing **Open in Safari** will open the Maps application.
- If a script contains a hide window command, FileMaker Go does not hide the window but changes the order of the open windows that are displayed.
- When you hibernate FileMaker Go, any currently performing scripts are aborted. If **Allow User Abort** is on, you can return to the previous state when you resume FileMaker Go. If **Allow User Abort** is off, FileMaker Go will close instead of hibernating.

Script Triggers

Script Triggers, including **OnTimer**, work in FileMaker Go as long as they use supported script steps. See the "Scripting for a Mobile World" section below for an example.




Calculations


The calculation engine is fully available on FileMaker Go with plug-ins being the only exception. Custom functions are also supported including recursive functions with the same stack limits as FileMaker Pro. Mobile devices do not have as much processing power, so performance needs to be considered when creating formulas that work with related data. Additionally, unstored calculations should also be used sparingly. This will be discussed more in the performance section. It is important to note changes to the following functions:


- **Get(SystemPlatform)** returns 3 when performed on the mobile device.
- **Get(ApplicationVersion)** for the iPhone and iPod Touch will return "Go 1.x.x" and for the iPad will return "Go_iPad 1.x.x".
- Given that there are none to press, **Get(TriggerModifierKeys)** and **Get(ActiveModifierKeys)** do not return expected modifier key information.
- **Get(WindowContentHeight)** and **Get(WindowContentWidth)** are the only two Window functions that change with device rotation.

Custom Menus

Custom Menus are supported in a limited way in FileMaker Go for the obvious reason that many of the menus that are available in FileMaker Pro are not available or applicable in FileMaker Go. You can modify and override the following menu items that are displayed in the interface:

- In the  menu in Browse mode:
 - Show All Records
 - Show Omitted Records
 - Omit Records...
 - Sort Records...
 - Enter Find Mode
- In the  menu in Find mode:
 - Show All Records
 - Show Omitted Records
 - Constrain Found Set
 - Extend Found Set
 - Perform Find
 - Exit Find Mode
- In the  menu in Find mode:
 - Add Find Request
 - Duplicate Find Request
 - Delete Find Request
 - Revert Find Request...

- In the  menu:
 - Add New Record
 - Duplicate Record
 - Delete Record...
 - Delete Found Records...
 - Revert Record...

- In the  Tools Menu Pop-over dialog:
 - Refresh Window
 - Save / Send

You cannot override the Quick Find menu item, the Save Record menu item, or the Exit Record menu item. You also cannot add custom menus to your databases nor can you remove menus or menu items. Menu items that are not included in your custom menus will still appear, but will be grayed out. Finally, you cannot use ampersands (&) in custom menu names.


Importing Records

Importing records is supported in a scripted process when using the Import Records script step to import from another FileMaker file. Using the Import Records script step, you can:

- Import records from a hosted file (FileMaker Server or shared using peer to peer sharing with FileMaker Pro and FileMaker Pro Advanced) into a local database. Although FileMaker Go cannot import directly from an ODBC source, it can import from an ODBC source through ESS.
- Import records stored in a local database into a hosted file
- Import records from a hosted file into another hosted file
- Import records from a local database into another local database
- Import records between tables in the same file (local or hosted)

Important Note: Field mapping and import order must be established before you import records with FileMaker Go. Creation of a new table during an import process is not supported. Please see the FileMaker Go Development Guide for more information.

Export Field Contents

Field contents can be exported in a scripted process or by clicking the Tools menu () in the upper-right of the application. Note that exporting multiple records is not supported: FileMaker Go will export only the contents of one field from the current (single) record. The resulting content can be emailed, viewed, or saved. If the content is saved, it will be available in the Files on Device list.

Consider a case where a text field contains a large amount of text. A user could export the contents and email them to a colleague in a few simple steps or through a scripted process.

Save Records as PDF

Save Records as PDF is supported as a scripted process or by clicking the Tools Menu (⚙️). As with exporting, the resulting file can be emailed and is available in the FileMaker Go Files on Device list accessible via the Window menu > File Browser option. It is important to note that while this feature is available, some of the script step options from FileMaker Pro are not supported:

- Appending records to an existing PDF
- Saving the current record, a set of records, or a blank PDF
- Selecting the Acrobat version
- Selecting Printing and Editing options on the Security tab
- Deselecting the option that allows screen reading software
- Options on the Initial View tab

FileMaker Go does not optimize PDFs for printing. For example, object sliding is not supported and choosing not to print objects is not supported. Displaying web viewers is not supported.

Save a Copy As...

New in FileMaker Go 1.1.1 is the ability to save a copy of a currently local database on the device. Saving a file as a compacted copy or a clone is supported when you use the Save a Copy As script step, but is not supported when using the menu item. The resulting file can be opened and used in FileMaker Go, sent via email or saved on the device where it appears in the Files on Device list. This feature can be useful for sharing files between FileMaker Go users or for storing a backup of a file. Imagine working remotely with a local database and receiving a request for access to your local file from your office. This feature would allow you to create and email a copy of the local database to those that need access. It is important to note that a copy of a local file is not linked in any way to the original file. Data that is entered into the original will not show in any copies of that file.

URL Schemes

Many FileMaker developers will be familiar with the `fmp7://<host>/<filename>` URL scheme that allows a database to be opened from a clickable link. This URL protocol has been extended to support script names and parameters. The new scheme syntax is:

```
FMP7Script://<host>/<file>?script=<script name>&param=<parameter>][&<$name>=<value>]
```

This update will allow FileMaker Go to communicate with other iOS applications. For instance, say you have an application on your mobile device that scans barcodes. Using a URL scheme from within FileMaker Go, you could launch the external barcode scanning application, scan an item and pass the barcode data back to FileMaker Go where the data could be used to lookup product information.

Script Step, Recognition of Runtime Files and Other Subtle Changes in v1.1

The Open Preferences script step now opens the FileMaker Go Settings screen (equivalent to tapping the Tool Menu > Settings), where users can set preferences for settings like the Status Bar display, **Auto-Correction** behavior, and **Auto-Restore Login** functionality.

FileMaker Go now supports the expected behavior for the “Go to next object using” layout field attribute when a user presses the tab or return keys. These field attributes are specified using the Inspector (FileMaker Pro 11) or Format Menu > Field/Control > Field Behavior... (FileMaker Pro 10 and previous). It is important to note that since the iOS keyboard does not provide an “enter” key, it is not applicable in FileMaker Go.

There has been a slight change to the “Record changed while disconnected” error message. The error “Your record changes cannot be saved because this record was modified by another user while you were disconnected.” now has a revert button so that users know what they need to do to get out of the current state.

FileMaker Go will now recognize files with an extension of “.USR” so that runtime solutions can be recognized and opened automatically by Go. A new file icon will be associated with the .USR files (the icon resembles the FileMaker document icon on Macintosh and Windows computers).

FileMaker Server

FileMaker Go can access External SQL Sources in databases hosted on FileMaker Server or shared via FileMaker Pro, through the use of ESS (External SQL Sources). ESS is supported just as it is with FileMaker Pro using the ODBC driver on the FileMaker Server machine or the FileMaker Pro desktop machine. Databases stored on the mobile device cannot access SQL Sources directly because there is currently no ODBC driver for iOS devices.

In FileMaker Server, “Go 1.x.x” or “Go_iPad 1.x.x” appear in the Admin Console.

There is no countdown timer in “Ask to Close” messages from the database host. A user will be prompted to close a file, and if there is no response after 30 seconds, FileMaker Go will close the file.

IPv6 is not supported by iOS.

What’s Out

Schema Modification

Developers cannot access the “Manage” menu options including the Manage Database dialog. All development for FileMaker Go databases must be done in FileMaker Pro or FileMaker Pro Advanced. This includes modifying tables, fields, security permissions, and relationships.

Layout Mode & Structure Modification

FileMaker Go has no Layout mode so developers cannot make changes to layouts or layout settings. Likewise, there are no editing capabilities for scripts, custom menus, custom functions, and so on. A value list of custom values can be edited so long as the “Allow editing of value list” option is selected when applying a field control style. Modification or edits to other value list types are not supported.

Note that this also prohibits gaining access to layouts that do not appear in the layout menu. Such layouts can be accessed in FileMaker Go using scripts or buttons for navigation.

Printing

The iPhone and iPad do not support printing. There is no Preview Mode on FileMaker Go, so the common practice of taking a user to a report layout, switching to Preview Mode, pausing, and then returning to the original layout after printing will need to be reworked for FileMaker Go. With the advent of live sub-summary parts it should no longer be necessary to take users to Preview Mode.

Charting

FileMaker Charts are not currently supported in FileMaker Go. As with FileMaker Pro prior to version 11, chart objects appear as a blank area in your layouts.

Exporting

FileMaker Go does not support record exports. Only field contents can be exported.

Plug-ins

Plug-ins are not supported.

Cocoa

FileMaker Go provides no access to the native programming environment for the iPad or iPhone—Objective C and Cocoa Library elements in native applications are not available to developers.

Hosting/Sharing

FileMaker Go does not allow sharing or hosting of any kind. If a file is local to an iOS device, only a single user is supported.

Save/Send Records As

Save/Send records as Excel and Snapshot Link are not supported.

Performance Considerations

Just as with FileMaker Pro, developers need to consider both network speed and operational speed when building databases for FileMaker Go. In testing there were few surprises. Performance with FileMaker Go is far more subject to its network connection than to anything else. Committing and creating records on a 3G network can take longer than expected, and sub-summary or sorting routines may be problematic. Generally speaking, operations that touch multiple records or modify keys that drive relationships should be carefully reviewed and tested for performance.

Performance Recommendations

Avoid long looping scripts, especially those that open and commit record data. If you need to work with large blocks of data, you might explore some of the strategies presented in the next section on scripting or choose to execute those routines that may be too slow on a mobile device in FileMaker Pro or as a scheduled script in FileMaker Server.

Be mindful of sub-summary reports and avoid large found sets. A list view with a large found set of records and one or more sub-summary parts can cause users to wait while the records are sorted and totaled. You might consider simplifying the list layout by removing sub-summary parts or use other means, like script triggers or navigational elements, to reduce the size of the found set.

Any images or graphics on your layouts should be compressed as much as possible.

Use portal sorting and relationship sorting judiciously. On a local area network, portal sorts can perform well, but on a wide area network with FileMaker Pro and FileMaker Go, sorted portals significantly increase the time it takes a layout to render.

In our benchmark tests, a MacBook Pro that was connected to the WAN via WiFi performed similarly to FileMaker Go on an iPhone and iPad. Setting aside the lower bandwidth available when connecting via 3G, the mobile devices only showed noticeably slower performance on those operations that required local processing power. It was interesting to note how completely some expensive operations are offloaded to the server. For example, one test involving a find on an unstored field in 350,000 records delivered almost identical results on a LAN connected Dual Quad Core Mac Pro,

a WAN connected MacBook Pro, and both WAN connected mobile devices. In contrast, an intentionally malformed recursive function displayed in a calculation field on a list of 300 records reached its recursion limit in under a second on the LAN connected Mac Pro, less than ten seconds on a WAN connected MacBook Pro, and took more than ten minutes to conclude on the iPhone and iPad. This being the case, be wary of processor-intensive tasks. In these same tests, performance degraded sharply on the 3G network.

This may be commonsensical, but use the excuse of less screen real estate to remove items from layouts. Scrutinize how many unstored calculations and related fields your layouts require, and be judicious with summary fields. All of these considerations are as true for solutions deployed to FileMaker Go as for those deployed to FileMaker Pro on a wide area network. This technical brief will not dwell further on WAN optimization except to remind you that it is important.

Tailoring an Interface for FileMaker Go

While you can simply open any existing FileMaker Pro database with FileMaker Go, the smaller screen and the touch interface of the iPhone and iPad will lead you to build layouts and workflows optimized for iOS.

Just as developers need accommodate the subtle differences in how text and objects are rendered on Mac OS and Windows, they will need to make sure to leave space for fonts and objects on iOS as well. There is some scaling that occurs, and you will want to test the appearance of your layouts on iOS just as you would for desktop platforms. This is particularly true for FileMaker Go as layouts are zoomed in single-integer increments—for example your screen might be zoomed to 64%—to fit on mobile screens. Because of this, objects and parts may not line up as cleanly as they do in FileMaker Pro at 100% zoom, 72dpi. Testing demonstrated some minor aesthetic differences. For example, you might see a thin line appear between list view rows, depending on their pixel row height, on your iPhone that you do not see in FileMaker Pro on your desktop. The key here, again, is to test your solutions for all platforms you intend to support.

FileMaker Go does support some gestures such as double-tap for zooming, pinch, and swipe to move around a zoomed screen. Since FileMaker Go also needed to ensure compatibility to existing FileMaker Pro databases, extensive use of new gesture-based features was avoided. One of the core strengths of the FileMaker platform has always been cross platform fidelity. With the introduction of FileMaker Go, experienced developers will need to learn the subtleties of the new platform, but FileMaker takes care of most of the differences automatically. For these reasons, developers do not have access to some of the touch interface gestures available on iOS.

Apple's Human Interface Guidelines for iOS are recommended reading. While many guidelines do not apply specifically to FileMaker Go, reviewing them can help focus a developer's thinking. You can find the Guidelines here:

For iPhone:

<http://developer.apple.com/iphone/library/documentation/userexperience/conceptual/mobilehig/>

For iPad:

<http://developer.apple.com/library/ios/#documentation/General/Conceptual/iPadHIG/index.html>

For Web Apps:
















<http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/>

Iconography

While no special restrictions exist for databases accessed with FileMaker Go, some standards do exist. These are both formal—from Apple's practices and recommendations—and informal on the iOS platform.

For example, it is a common practice to use a plus "+" icon to suggest adding something to a database or to display a picker mechanism. A ">" icon, often found in a circle, denotes displaying a detail view. Trash cans, a long-time staple of FileMaker developers, usually indicate deletion as do circles with slashes or "x" icons. A magnifying glass suggests search functionality, and a curved arrow pointing upwards usually implies sending data to someone or something else.

Table 10-1 from the iPhone Human Interface Guidelines shows some of the standard buttons the iPhone platform provides for toolbars and navigation bars.

Button	Meaning	Name
	Opens an action sheet that allows users to take an application-specific action	Action
	Opens an action sheet that displays a photo picker in camera mode	Camera
	Opens a new message view in edit mode	Compose
	Show application-specific bookmarks	Bookmarks
	Display a search field	Search
	Create a new item	Add
	Delete current item	Trash
	Move or route an item to a destination within the application, such as a folder	Organize
	Send or route an item to another location	Reply
	Stop current process or task	Stop
	Refresh contents (use only when necessary; otherwise, refresh automatically)	Refresh
	Begin media playback or slides	Play
	Fast forward through media playback or slides	FastForward
	Pause media playback or slides (note that this implies context preservation)	Pause
	Move backwards through media playback or slides	Rewind

While these icons will be familiar to iPhone and iPad users, there is a risk in too literally recreating an iOS-like interface. FileMaker Go behaves differently than native applications, and it will be important for users to remember that they are still working with connected databases within the FileMaker paradigm. Likewise, FileMaker Pro users on desktops may be put off by having to manipulate an interface exclusively designed for iOS devices. Still, it makes sense to leverage user expectations and create interfaces that are intuitive and common to applications with which they are already familiar.

Log Out Buttons

During testing with FileMaker Go, only the most disciplined users remember to close all open files before hitting the Home button and closing FileMaker Go. Most users will, when finished with FileMaker Go for the moment, immediately hit the device Home button without a second thought. If your database requires a more graceful exit, you cannot ensure or enforce a different behavior, but you might consider placing a large “Log Out” or “Exit” button at the top of your interface screens in the hope that users can be trained to use it.

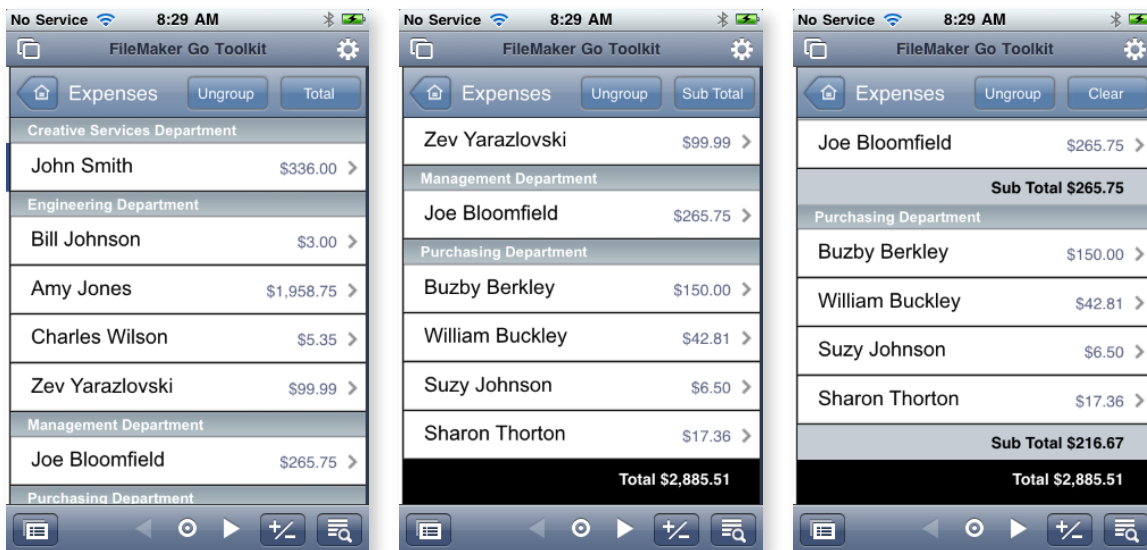
Refresh / Request-based Processing

Mobile device users will generally be comfortable tapping a refresh icon to retrieve fresh data from a server. This is a common element in many connected and web-based mobile applications. This opens a world of possibilities for FileMaker developers. While many FileMaker Pro users, especially those used to working on a local area network, would not expect refresh functions, mobile users will not only accept this user interface practice, they can benefit from it. Users can choose explicitly when they are able to wait for data and performance-intensive processes to complete and explicitly request them.

For example, imagine a database that shows a list of employee expense reports and a subtotal. In a standard sub-summary list view, FileMaker Go would redraw the list and recalculate the totals each time a user viewed that layout. This might create a performance delay for mobile or wide area network users.



Given that the idea of refreshing data on a mobile device is more familiar to mobile users than it might be to FileMaker Pro users, a developer could create a list view that does not use live sub-summary functionality, but rather relies on stored totals and updates those totals when the user explicitly requests new data. Note the controls added to the top of this example for grouping data and displaying totals.



Adopting this sort of mechanism clearly depends on how developers approach their database and on the expectations of their users. This is one example of how a developer might consider adjusting an interface to accommodate both performance issues and the different experiences with mobile devices. An argument can be made that this practice would benefit FileMaker Pro users across wide area network connections as well.

Sizing Layout Elements

On the iPad and iPhone the pointing device—a human finger—is less precise than the pointer for a mouse. As such, you may want to adjust the size of your layout objects, including buttons, field heights, and the row height for list views and portals, accordingly.

15 pt (17px)

16 pt (18px)

17 pt (19px)

18 pt (20px)

20 pt (22px)

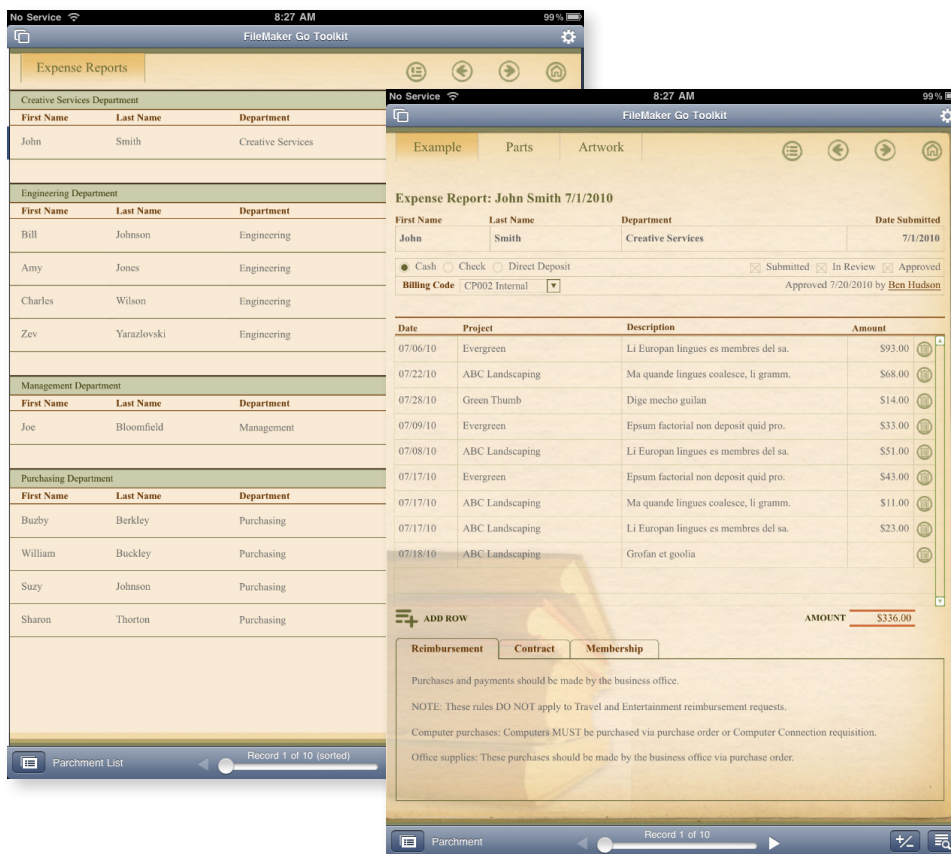
22 pt (24px)

In Apple’s Human Interface Guidelines for the iPhone, Apple recommends using fonts between “17-pixel and 22-pixel.” In FileMaker terms, using Helvetica on Mac OS X, the font’s vertical pixel dimension is generally two higher than its point size. In the screenshot below there is a gray box shown behind the font that is sized to the point size of the font in each row.

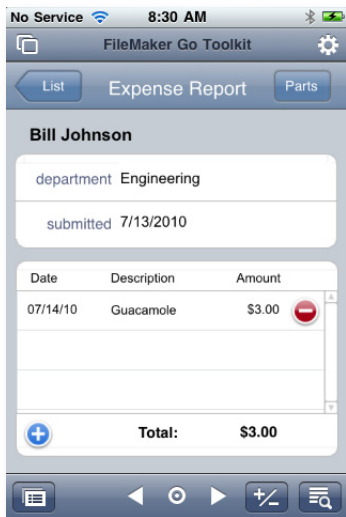
In practice, you can generally make labels smaller than buttons and field text. Since users do not interact with labels, the size of a human fingertip does not matter as much as it does in situations where a user needs to edit or click the text.

Likewise, row height— for both portals and list views—should be between 34 and 42 pixels high.

Lastly, button hit areas should be a minimum of 32 pixels square and can be as large as 52 pixels high to conform to what most users on the iPhone or iPad are used to.



The example above demonstrates what might be considered a reasonably well laid out interface for the iPad with a form view and a list view. The example also shows a JPG background image that has been compressed significantly.



The example above mimics Apple’s guidelines for an expense report layout that might appear on the iPhone.

Please note that these are only recommendations. Interfaces should be driven by the needs of the people using a database. It is not the intention of this technical brief to introduce new dogma into the development community.

Likewise, interfaces primarily aimed at iPad and iPhone users may be completely inappropriate for users of FileMaker Pro on the desktop. One of the core strengths of the FileMaker platform is its ubiquity and compatibility; one layout can serve the needs of Mac OS users, Windows users, and iPhone/iPad users. Using pinch-zoom and swipe scrolling, you can create a common ground for your users without alienating desktop users by exclusive use of iOS-like layouts.

Finally, we should emphasize that for casual or occasional use, existing interfaces may work just fine. In the same way that most people use their iPhones to successfully browse desktop-sized web sites, they can use FileMaker Go to move around a FileMaker Pro interface. Device-specific-interfaces could be easier to use and improve mobile productivity, but you should weigh the benefits of separate touch-specific or size-specific screens versus the effort required to create and maintain them.

Object Autosizing & Portrait/Landscape Rotations

If you are not already making heavy use of the automatic sizing of layout objects in FileMaker, available via the Anchor settings on the Position tab of the Inspector, developing for FileMaker Go should make it a necessity. Users can rotate their devices from landscape to portrait at any time, and FileMaker Go does an excellent job of redrawing the current layout accordingly. Keep in mind that graceful resizing of layouts depends on your minimum dimensions being the smallest common denominator.

Screen Dimensions

For reference purposes, these are the screen dimensions for FileMaker Go in various states:

Remember: List View consumes an additional 3px of width for the selection indicator on the left. This difference is not reported by `Get(WindowContentWidth)` and it is not taken into consideration in this matrix:

	Portrait		Landscape	
	Toolbar	No toolbar	Toolbar	No toolbar
iPad	768W x 929H	768W x 973H	1024W x 673H	1024W x 717H
iPhone	320W x 385H	320W x 429H	480W x 255H	480W x 289H

In order to test for whether a user has their device in landscape or portrait mode, and to determine how much screen real estate you have available, use `Get(WindowContentWidth)` and `Get(WindowContentHeight)`. Note that regardless of your zoom level, the values above are what FileMaker Go will report.

Window Management

FileMaker Go behaves similarly to FileMaker Pro on the Windows operating system in maximized mode. Only one window displays at a time in FileMaker Go. Mobile users cannot position two windows on their display next to each other and click between them. Windows are managed in a separate interface that mimics the way Safari works on the iPhone and iPad.

There is no benefit to hiding a window on FileMaker Go; a hidden window behaves the same as a non-hidden window. Floating (smaller) pop-up windows or pseudo-modal dialogs take over the whole screen. Off-screen processing windows are not drawn off screen; they are fully visible while they have focus. If you have a script that relies on an off-screen window, your users will see FileMaker Go flip to that window where it will remain on screen until your script switches focus back to the original window. For some developers this may significantly impact their scripting habits.

All of this leads to the general recommendation that you should design your solutions to be single-window-centric in their primary functions.

Zoom Management

Window zoom settings behave somewhat differently in FileMaker Go. Zoom settings can be applied to a layout and those settings will be retained for the layout until the setting is changed. This is slightly different than in FileMaker Pro, where the zoom setting is applied to the window and is retained even when a layout change occurs.

Additionally if you script a zoom level that is smaller than the zoom level that will fill the screen, FileMaker Go stops at the fractional zoom value that lets you see the whole layout, but no smaller. FileMaker Go will not let you zoom out to see a postage-stamp version of your layout; it will only go as far as the screen allows.

For example, if you view a layout that does not fit the screen and pinch it, it will not get smaller than what can fit on screen, say 73%. If you use a script to set zoom level lower than 73% in this case (say to 50%), FileMaker Go will only zoom to 73%.

Note that `Get(WindowZoomLevel)` returns the integer value of a screen, even though FileMaker Pro users and scripts can only change the zoom level in set increments of 25%, 50%, etc.

Locking the zoom level can be helpful on a list view to prevent horizontal scrolling. In general though, given that users will more dynamically control zoom levels by pinch-zooming whenever they feel like it, a possible best practice would be to defer zoom levels to the user rather than setting them by script as is common in FileMaker Pro.

Scripting for a Mobile World

As mentioned above, one of the important considerations to keep in mind for FileMaker Go is that a script can be halted at any time regardless of its Allow User Abort settings. If a call comes in on the iPhone, iOS closes the existing application to take the call. FileMaker Go does not support multitasking, so the application is shut down when that happens. Likewise, a user can close the application at any time by pressing the Home button on either the iPhone or the iPad.

Performance is another important consideration. While it is possible to create some fairly elaborate and richly scripted processes in FileMaker, the addition of FileMaker Go to the platform further stresses that developers should be judicious when using “cool pet tricks” or data intensive processes.

Testing indicated that committing records is a more network-intensive function than most other operations in FileMaker Go, such as performing a find request. If, for example, a developer were to create a looping script that modifies a field in a found set of records, two potential issues may arise. Firstly, the Home button, a phone call, or a dropped connection may interrupt your script; and secondly, your users may not be prepared to wait for the results, especially when on a 3G network.

A poorly conceptualized script in this case could leave a significant portion of your data in an unmodified state. It cannot be stressed enough that in FileMaker Go there is no guarantee a given script will finish. A strong best practice is to be careful and deliberate when writing scripts that modify data.

Before being guilty of a Chicken-Little response to this reality of working in a mobile world, consider that it would be rare for someone “in the field” to need to run large batch scripts. While a concern, and something this technical brief addresses, the practical reality is that developers will accommodate the limitations of mobile networks. Developers should consider leaving the heavy lifting to FileMaker Pro users and FileMaker Server.

Scripting Strategies

There are a few different ways to approach data processing scripts for mobile or, more universally, WAN-accessed databases. The following techniques are meant to illustrate the ideas without necessarily suggesting that these are the only or universally correct approaches.

Testing for Platform

There is no additional permission required to open a database in FileMaker Go; this is an important subtlety to understand. If users have permission to open a file in FileMaker Pro, they can also open the file in FileMaker Go. Given that complex database solutions are likely to contain scripts that need to be tested before being used in FileMaker Go, you may want to prevent mobile users from connecting to your database until you have checked for compatibility.

As mentioned above, you can test which platform someone is using by using the calculation function **Get (ApplicationVersion)**. The iPhone and iPod Touch will return “Go 1.x.x” and the iPad will return “Go_iPad 1.x.x”. You can also use the **Get (SystemPlatform)** function which will return a result of “3” when evaluated from a mobile device using FileMaker Go.

If you add an If step to a script set to run when your database opens, you can effectively prevent mobile users from accessing the database:

```
If [ Get ( SystemPlatform ) = 3 ]  
    Show Custom Dialog [ Title: “Mobile Access”; Message: “This database does not yet support access by FileMaker Go. Please contact your developer for further information.”; Buttons: “OK” ]  
    Close File [ Current File ]  
End If
```

OnTimer Scripts

OnTimer triggered scripts are set to run at specified intervals. A key difference between a script set to execute with an OnTimer trigger and any other script is what happens when a user hits the Home button or a call comes in and forces FileMaker Go to close. A script being triggered with the OnTimer script step is smart enough to know how much time has elapsed even though the application may have been closed. Once the application is reopened, the OnTimer script will know if the interval has passed and will trigger the script in the event that it has.

For example, if you managed your daily appointments using FileMaker Go and had a reminder set to alert you 10 minutes before an appointment’s start time, you could use an OnTimer script to help FileMaker Go keep track of your reminders even if a call comes in. No other trigger or scripted process can continue in the event that FileMaker Go is closed. In addition to the OnTimer script, you need to make sure that you add an **Allow User Abort [Off]** script step in your script. If you do not add an **Allow User Abort [Off]** script step, then the script will abort, and the OnTimer script will cease to function until you are able to trigger it once again. This technique demonstrates one of many possible uses for an OnTimer trigger using FileMaker Go.

Using Location Information

While FileMaker Go cannot access any GPS features of the iPhone, with a little creativity and a web viewer, it is possible to work around this limitation. The technique involves a combination of a simple HTML5 page and some javascript. Since Safari on the iPhone supports HTML5 geolocation services, so can FileMaker Go because a web viewer object uses the Safari engine to handle rendering of web pages.

Imagine a sales team that could “stamp” their customer locations while doing data entry along a travel route—all within FileMaker. Likewise, it would also be possible for a user to record mileage (at least as the crow flies). There are numerous possibilities.

Chad Novotny (The Support Group) and Todd Geist (Geist Interactive) have each posted blog entries that show how to accomplish this. A Google search will quickly provide links to both articles.

Scripts That Run On Open and Close of a FileMaker Database

Scripts set to run when a database opens and closes behave the same way as they do in FileMaker Pro, but the user behavior and FileMaker Go’s hibernating of FileMaker Go require extra consideration of these scripts. As with FileMaker Pro, an opening script will run when a file’s first window is displayed. This is true for FileMaker Go. However, if a user returns to FileMaker Go after closing the application and responding with a “yes” to the dialog asking if he wants to resume working with his prior open databases, FileMaker Go will do exactly that—return the user to his prior state and not perform any script specified in the File Options.

Likewise, scripts set to run when the file closes only run when a user (or a script) closes the final open window for a given file. If a user hits the Home button or is interrupted by a phone call, FileMaker Go will close without running any additional scripts including the one set to run when a file closes.

Related Record Locking Transactions

While not exactly a scripting strategy, a consideration is the potential for related records to get out of sync in dependent edits. If, for example, you needed to ensure that either all changes to a related set of records are committed or none are, one approach is to use the natural behavior of the locking mechanism in FileMaker Pro. If you open a record, any changes made to related records through portals while that record is uncommitted are also uncommitted.

Todd Geist, a regular contributor to the FileMaker developer community, has spoken and written about a transactional model that takes advantage of this fact if you need to ensure an atomic operation on a complex process. All changes to a parent record, and to its children made through portals on the uncommitted parent, will either commit or completely roll back if the operation gets interrupted. This technique has been around for years and is well understood by advanced developers. It is of renewed interest given the much less persistent connection to the host from mobile devices.

Read the FileMaker Advisor article <http://my.advisor.com/doc/17403> (requires a subscription), or Todd has posted some information at <http://www.geistinteractive.com> and <http://www.geistinteractive.com/content/inventory-transactions>.

Server Side Queuing

Rather than having FileMaker Go handle your entire scripting load, consider moving some scripts to your FileMaker Server or a FileMaker Pro client machine dedicated to supporting your solution. Queuing is a common technique used by FileMaker developers. It is sometimes referred to by different names; asynchronous processing, (ro)bot processing, and server side processing all refer to this concept. Namely, a user does something that puts a request into a queue, which then gets processed by a daemon or scheduled process running at some regular interval on the host or robot/client machine.

This technique is frequently used for tasks that may take a lot of processing time, and therefore would block the user from doing other things if performed on the client. This technique has new relevance in mobile apps. Mobile devices have less

processing power and more intermittent connections, so task queuing might be worth considering in more situations than it normally would have been in solutions designed only for FileMaker Pro on a local area network.

Queuing works particularly well when you can have timely “push” notification, and the **OnTimer** trigger makes a great compliment to this technique. Note that **OnTimer** script triggers will only work when FileMaker Go is actively running.

Here’s how the technique works:

1. User initiates your script by tapping a button or running a script from the scripts menu.
2. Rather than process the script there on the FileMaker Go client, your script creates a record in a “queue table” with the name of the script intended to run, the user who requested it, and any pertinent script parameters or required data.
3. Either a FileMaker Pro client or FileMaker Server running its own “watcher” script on a timer notices that there is a new script request in the queue table. Based on the name of the script and the parameters provided it runs the batch process.
4. As a nice way to notify the user that their script is done, an **OnTimer** triggered script on FileMaker Go could watch the queue table and note when a record is listed as being complete. It could then show a dialog to the user informing them of this.

This is by no means the only way to work, and glosses over some challenges with context and found sets, but it is an approach worth considering.

Script Logging / Flagging

Another approach to ensuring that data is not left in an incomplete state by an unfinished script is to simply test for a result condition. This idea could be implemented in a number of ways, but the essential idea is to flag each record modified by a script with a timestamp or some similar element and then to, perhaps via an **OnTimer** script, periodically look for records missing that timestamp. Your database could then ask the user if they wish to run the script again in order to conclude its process.

Common sense should dictate your approach to these issues. If you write a script that, for example, generates a report for the user, no one will care if a connection drops and the report is not presented. Your data will be fine and the user, once she reconnects, will have an opportunity to run the report again.

The intent here is to be mindful of the issues around halted scripts and plan accordingly.

Appendix A: Unsupported Script Steps (excerpted from the Developers Guide)

FileMaker Go does not support some script steps.

Running scripts without handling unsupported script steps appropriately may lead to unintended behavior or data corruption.

Returns Error code 4

The following script steps return error code 4: Command is unknown. Use **Get(LastError)** to check for this error code. If **Allow User Abort** is on, FileMaker Go displays an alert that gives the user the choice to continue or to abort the script. If **Allow User Abort** is off, FileMaker Go displays an alert informing the user which script step is unsupported unless **Set Error Capture** is on, which prevents any error display dialog from appearing. Additionally, users cannot abort scripts if **Allow User Abort** is off.

Category

- *Script step*

Editing script steps

- Perform Find/Replace

Fields script steps

- Insert from Index
- Insert Picture
- Insert QuickTime
- Insert File

Files script steps

- Print
- New File

Miscellaneous script steps

- Execute SQL

Navigation script steps

- Enter Preview Mode

Records script steps

- Import Records (import between FileMaker files is supported)
- Export Records
- Save Records as Excel
- Save Records as Snapshot Link

Returns Error code 3

The following script steps return error code 3: Command is unavailable. Use `Get(LastError)` to check for this error code. FileMaker Go does not display an alert for these script steps.

Fields script steps

- Insert Object (Windows)
- Update Link (Windows)

Files script steps

- Print Setup
- Set Multi-User
- Recover File
- Convert File

Miscellaneous script steps

- Allow Toolbars
- Speak (Mac)

- Send Event (Mac)
- Send Event (Windows)
- Perform AppleScript (Mac)
- Send DDE Execute (Windows)

Open Menu Item script steps

- Open Edit Saved Finds
- Open File Options
- Open Manage Database
- Open Manage Data Sources
- Open Manage Layouts
- Open Manage Scripts
- Open Manage Value Lists
- Open Find/Replace
- Open Sharing

Spelling script steps

- Check Selection
- Check Record
- Check Found Set
- Correct Word
- Spelling Options
- Select Dictionaries
- Edit User Dictionary

Windows script steps

- Show/Hide Text Ruler
- Move/Resize Window
- Arrange All Windows

Open Help is a supported script step; it will open the web view to the help page, but will not exit the application.

Knowledge Base Articles

Please also check out the Knowledge Base articles on the FileMaker website at:

<http://help.filemaker.com/>

Credits

Written by Scott Love and Aaron Gutleben at Soliant Consulting with assistance from Jeremiah Small and Carl Young.



Soliant Consulting creates custom databases, web sites, and mobile applications for businesses around the country. We specialize in FileMaker, PHP and the Zend platform, the iPhone/iPad OS, Adobe Flex, and Salesforce.com.

<http://www.soliantconsulting.com>

©2010 FileMaker, Inc. All rights reserved. FileMaker is a trademark of FileMaker, Inc., registered in the U.S. and other countries. The file folder logo is a trademark of FileMaker, Inc. All other trademarks are the property of their respective owners. Product specifications and availability subject to change without notice.

THIS DOCUMENT IS PROVIDED «AS IS» WITHOUT WARRANTY OF ANY KIND, AND FILEMAKER, INC., DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, OR THE WARRANTY OF NON-INFRINGEMENT. IN NO EVENT SHALL FILEMAKER, INC., OR ITS SUPPLIERS BE LIABLE FOR ANY DAMAGES WHATSOEVER INCLUDING DIRECT, INDIRECT, INCIDENTAL, CONSEQUENTIAL, LOSS OF BUSINESS PROFITS, PUNITIVE OR SPECIAL DAMAGES, EVEN IF FILEMAKER, INC., OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY. FILEMAKER MAY MAKE CHANGES TO THIS DOCUMENT AT ANY TIME WITHOUT NOTICE. THIS DOCUMENT MAY BE OUT OF DATE AND FILEMAKER MAKES NO COMMITMENT TO UPDATE THIS INFORMATION.