

Offscreen Windows

Before to start

As emerged by discussing this topic, the best practice about offscreen windows is to simply avoid them if you can. So this page attempts to list the best practices to use when you just have to use offscreen windows. Following there is a list of pros and cons of offscreen windows as emerged in the discussion:

Cons:

- Performance: Freezing the window and switching layouts is faster than opening a new window
- Method not compatible with FileMaker Go
- For Windows users with the window maximized, opening a new off-screen window will un-maximize the previous window
- You have to remember closing them at every script's exit point

Pros:

- It's an easy method to preserve the application state (current layout, found record set, etc.)
- Makes scripts more context free

Offscreen Windows

Every time a script is supposed to do something more than just update a couple of fields in the current record, I like to make it work in its own window (often off-screen to improve the user experience). One problem with the windows is that you have to remember to close them at every exit point of your script, otherwise your solution will accumulate tens (at best) of off-screen windows. I'm using a couple of practices that I found helpful in the everyday work:

- Set the title of the window to the name of the script that is creating it. This helps in the debug process when you find one (or many) offscreen windows left open by scripts; moreover it makes very easy to close the right window at the end of the script (by specifying the script name as the title of the window for the close command), this greatly reduces the risk of conflicts in the window usage because every script, also if it is called as a subscript, will always open and close its personalized window. The only case in which this method needs additional precautions is the event of recursive script calls, in this case the script should be able to reuse an open window (for resource usage efficiency) or open a specific window for every running instance of the script (for example by appending a random number at the end of the window title to uniquely identify it).
- I created a script called "Create offscreen window (title)" which I call in my scripts when I need a working window for them. In this script I create an offscreen window with the specified title; if the current user is a developer the window is not placed offscreen for debug purposes. This script gives me a cleaner perception of the solution because I have this explicative script called instead of hundreds of Create new window commands around in my scripts. As far as you avoid or manage recursive calls, this script can be modified to recycle existing windows when the specified title matches an existing window's title.

These are practices that I'm constantly using but they are far from refined; for instance I was thinking about a custom function to generate a unique script instance identifier (something like script name & timestamp & random number) that could be useful for debug/log purposes (with recursive script calls too) and also usable as a title for their working windows, but at the moment the techniques I described above are enough for me.