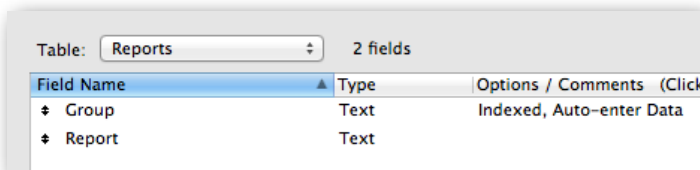# PRIVILEGE SET REPORTING BY MICHAEL ROCHARDE

FileMaker has always been an incredible tool for generating reports and for many years, I used to have a report screen with banks of buttons to generate all of the individual reports. In the last few years however, I've added a global field, to the footer of each module data entry screen, with a pop-up menu, listing all of the different report options for the module in question.   To that global field, I've added an **onobjectmodify** script trigger which sets the name of the report as the parameter that generates the report they want. Simple, very effective and it has the added advantage of giving the user all the reports that are relevant to where they are at the time.

Recently I had a project where some of the reports that were required were restricted to the different privilege sets and I was looking for a quick (and easy) way to accomplish this. I started thinking about whether I could use a related value list to solve the problem and came up with, what I think is, an elegant and very simple solution.

The first thing I did was create a **Reports** table with just two fields, **Group** (for the privilege set) and **Report** (for the name of the report itself).
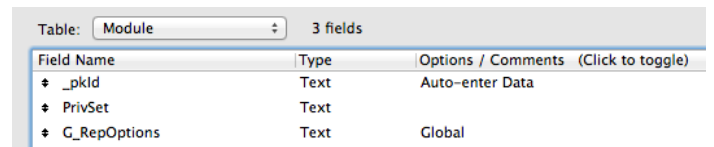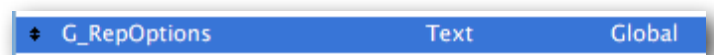

The next thing was to create the Privilege Sets themselves
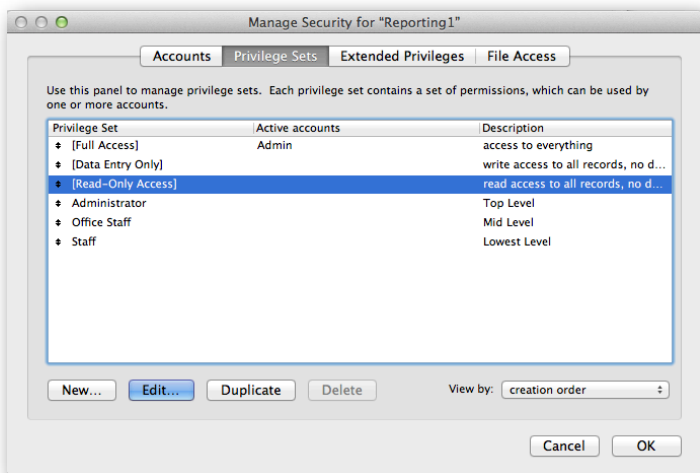
I then created a global field, **GRepOptions**

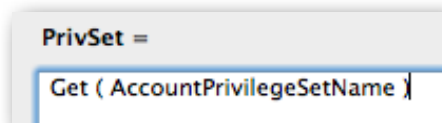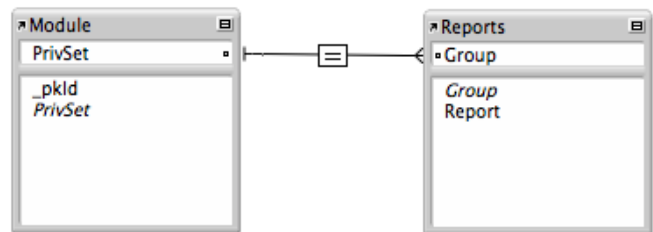(which is going to display all of the different report options) in my **module** table

in which I have a calculated field **PrivSet**

which is used in the relationship to restrict access to certain reports

(Note that this table is just for demonstration purposes but, in the real world, you would add the **GRepOptions** field to all of the tables that you want to report on).

We'll now add our **reports** layout to list all of the different reports and add access to each of them. *(For this technique to work, you must name the layouts exactly the same as the name of the report it is running)*

As you can see, the **Administrator** group has access to all **6** reports, **Office Staff** has access to **4** reports and **Staff** only has access to **2**.

The **Group** field

uses a value list that has the different privilege sets.

which is all very well and good but, as a developer, I need to have access to all of the reports and the relationship doesn't allow that as I'm logged in as **"[Full Access]"**. For this reason, I set an auto-enter option on that field for each new record:

(Note that the user doesn't see this value and is not aware that it exists nor can they change it).

All that we have to do now is add our **GRepOptions** field to our screen and format it to display the related reports
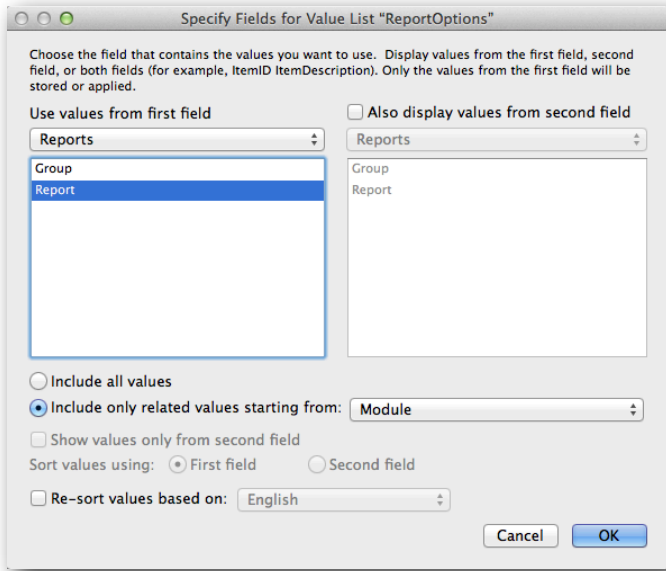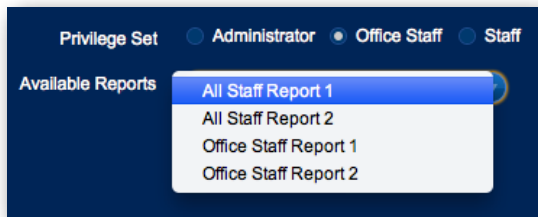

So with all done, let's look at what our options are for each of the different privilege sets:
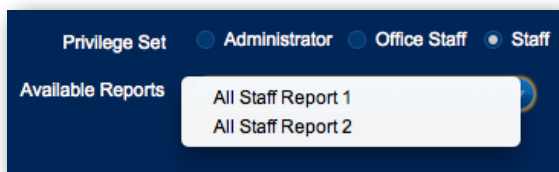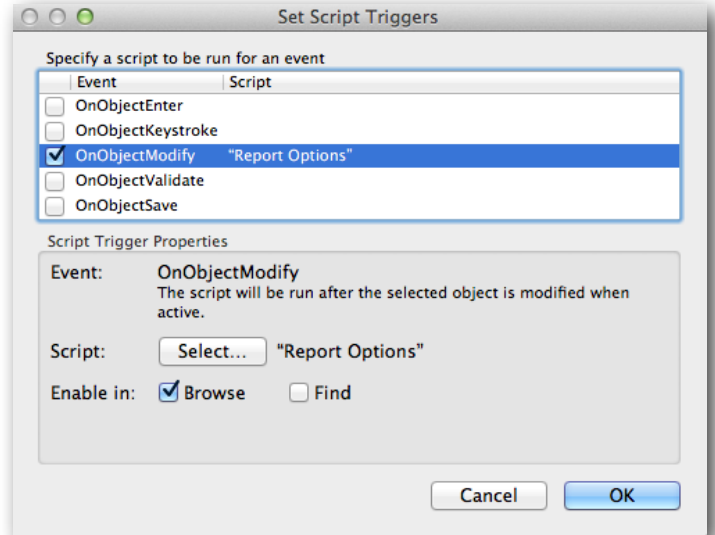
Administrator
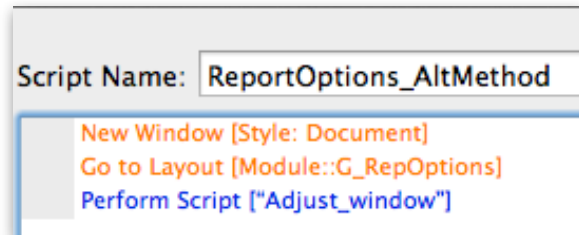

Office Staff

Staff

*(Bear in mind, that on your screen the **PrivSet** field doesn't need to be displayed. If you do want it to show, I would insert it as a merge field on the layout and position it just above the pop-up menu field.*

We're almost done. We now just need to add a script trigger to the **GRepOptions** field:

which runs the **Report Options** script

and that's it.

Just as a recap, the only reports that each user can choose from are the ones available for their specific privilege set. We don't have to set up any complicated privileges or access or write lengthy conditionals, in the script, to restrict access.

Because you're using the value of G_RepOptions to select which layout to use, *(A big thanks to Doug West of Excelisys for pointing this out),* you don't even have to type the layout names which makes it a really easy to implement and use.

(If you wanted to, you could add a simple integer number field to each report and then have a number field (in your reporting table) which looks up the value of the currently selected report and then uses that value as the script parameter).

One Last Thing

All of the work that you've done here could easily be undone if all users could access the Reports screen. For that reason, that layout is hidden and can only be accessed via a button that runs the following script:

```
Script Name: Reporting Options

If []
Else If [Get(ScriptParameter)="reporting" and (Get ( AccountPrivilegeSetName )="[Full Access]" or  Get ( AccountPrivilegeSetName )="Administrator")]
    Go to Layout ["Reports" (Reports)]
    Perform Script ["Adjust_window"]
Else
    Show Custom Dialog ["Access Restricted"; "Sorry but you do not have access to this function."]
End If
```

The Else If line is:

Get(scriptparameter)="reporting' and
(Get(AccountPrivilegeSetName) = "[Full Access]" or
(Get(AccountPrivilegeSetName) = "Administrator")

Note: There is an extra set of parentheses surrounding the second part of this line; this allows you to only use get(scriptparameter) once. (Again thanks go to Doug West for pointing this out).

This means that only developers and Administrators can access the screen to add/delete reports and set privileges.

As I said at the beginning of this article, this is a very simple solution. Sometimes, as developers, we get so wrapped up in what we're doing that we make things too complicated; just because we can.

Always remember **KISS** (keep it simple, stupid) and you'll be amazed at how easy some problems are to solve.

About the Author

Michael Rocharde has been a professional FileMaker Developer since 1987. He has worked with Excelisys (www.excelisys.com) since 2002 and has had more than 40 articles published in a variety of magazine

He is also the author of **FileMaker & Me**, an interactive multimedia book focusing on FileMaker Interface Design. Available only for the iPad, this book is available at the iTunes store:

itunes.apple.com/US/book/isbn9780985366100.

Michael recently released a free reference guide to FileMaker 12's Themes (**Themes Ain't Wot They Used to Be**) (http://www.excelisys.com/blog/2013/03/08/free-ebook-on-filemaker-pro-themes-by-michael-rocharde/)

Currently Michael is working on a new book '**FileMaker & You', Tips & Tricks for Successful FileMaker Development**' (To be published May 2013) for beginners to intermediate developers who, typically, are working in-house and who are being tasked with building a FileMaker solution to manage a specific task or function. Often these people are thrown into the deep end and expected to come up with a perfect solution for the problem, without having any of the tools, or the experience needed to actually succeed. Because of this, many projects are doomed to either fail or never work quite well enough to be successful.

Michael lives in the south of France, just outside of Toulouse in a small village called Castelmaurou. When he's not developing FileMaker solutions or writing, he can be found tearing out what little hair he has left trying to master the French language.

He can be reached via email at:

rocharde@excelisys.com

or via Skype **michael_rochard** or at **360-450-5583**

excelisys