



<http://www.nightwing.com.au/FileMaker/>

CobaltSky@nightwing.com.au

FileMaker Pro™ Dynamic Portal Sorting

Paper prepared by R J Cologon PhD

August 2002

Originally published as a series of posts on the FileMaker Café online forum, this paper provides a brief and practical overview of the techniques for creating dynamic sorting of portals in FileMaker Pro versions 4.x to 6.x.

Before getting straight into the technique, let's first note that it is assumed that you have a good broad understanding of FileMaker Pro development techniques, and that you have considered:

- Why you might want to use this approach rather than any of several others, and
- What the implications of such an approach are, and in what circumstances it might not be appropriate - for instance, scrolling portals can become unwieldy when very large amounts of data are involved - and portal sorting in such cases may not be responsive enough

It is also noted that there are several excellent products available which facilitate portal sorting and/or filtering. It is not possible to discuss all cases here, so it is left to the good judgement of the reader to determine whether the techniques discussed here are suitable for a particular application — and when an alternative solution might be more appropriate

The Basic Technique

Here's a fairly straightforward method to achieve dynamic re-sorting of related values (ie in a portal), achievable in four easy steps:

1. Create a value list in the main file called 'Sort_Key' which lists the names of the different fields in the related file that you wish to be able to sort by.
2. Create a global text field in your related file called 'gSortKey', place it on the layout in your main file above your portal, with a label that reads 'Sort related records by:', then go into 'Field Format' and define gSortKey field as a pop-up menu based on the Sort_Key value list.
3. Create a calculation field in your related file called DynamicSortValue, with the formula: GetField(gSortKey).
4. Define the relationship in the main file to sort on the 'DynamicSortValue' field.

That's it - you're in business. The gSortKey in your main file layout will operate as a menu of sort options and the portal will re-sort dynamically when you select a new option.

If desired, buttons may be placed beneath column headings above the portal (instead of the suggested sort menu) and linked to scripts which will set the 'gSortKey' field — to provide click-to-sort functionality.

Please note:

- A) The style of sort (ie number or text) will depend on the calculation result type you specified for the 'DynamicSortValue' field - and that should match the data type of the fields in the related file that you're including on the gSortKey menu.
- B) If you want to list both number and text fields in the sort menu, you'll need both a number and a text calculation field in the related file (with complementary formulae based on the 'Case' function), and a secondary key in the sort definition for your relationship.
- C) If you should re-name any of the fields included in the sort, you will have to remember to manually update the Sort_Key value list in the main file.
- D) Multi-key sorts or reversal of the sort order based on the same approach require a couple of extra steps.

More complex dynamic Portal Sorts

For those who may wish to go further than the basic sorting technique described above, here are some methods to achieve more complex dynamic sorting options in portals.

To achieve a reverse-sort option, a global number field checkbox (valuelist defined as 1 or 0) called 'gSortOrder' should be added alongside the gSortKey field. Two calculation fields are then required in the related file, with the formulae: `Case(not gSortKey, GetField(gSortKey))` and `Case(gSortKey, GetField(gSortKey))` respectively. Both fields should then be included in the sort options defined for the portal relationship, with the first defined as an ascending sort and the second as descending.

To achieve sorting on a multiplicity of data types, pairs of calculation fields must be created, one for text, and another to apply number sorting to all other field types (time and date require a number sort in FileMaker Pro). The fields must be defined with arguments which selectively activate them when a sort key is selected which is of an appropriate type. The formulae for this purpose may be defined along the lines of:

```
Case(PatternCount(FieldType(Status(CurrentFileName), gSortKey) , "Text"),  
GetField(gSortKey))
```

```
Case(not PatternCount(FieldType(Status(CurrentFileName), gSortKey) , "Text"),  
GetField(gSortKey))
```

In this instance, both fields should be included in the sort defined for the relationship. If a reverse sort option is to be included, then a second iteration of each of the fields will be required (defined along the lines outlined for a single reverse sort option) and all four fields will form part of the portal sort definition (two ascending and two descending).

Finally, to achieve a multi-key option an additional global sort key field should be defined for each successive key, and the whole process repeated for each key (with the calculated fields for the second sort key being added to the portal sort order after the group of fields for the first key and so on).

Thus to achieve a three-key sort on four different data types with a reverse-sort option, four global fields and twelve unstored calculating fields are required in the

related file. It works just as well as the simplified version described as the 'basic technique, above, but needless to say, takes a little longer to implement.

If value lists are to be used for sorting it is desirable that a 'null' option be included in the Sort_Key list in the main file when a multi-key sort interface is being defined, so that users are not obliged to specify all available keys. An 'unsort' button (linked to a script which clears the global sort key fields) is also a nice touch.

Refreshing Issues

Simply changing the values of unsorted calculating fields in the related file is not enough, in and of itself, to prompt resorting of the values in the current portal. To ensure that the portal will refresh immediately, a refresh script is desirable.

When there are many records displayed in a portal (more than can appear in the portal at one time), a simple script with the steps:

- ‡ Go To Portal Row [Select, Last]
- ‡ Exit Record/Request

will prompt refreshing of the portal display, and the new sort order will be applied. In some cases, shorted portals (ie with fewer related records than the number of rows of the portal) will not be properly refreshed with this simple approach. In this case, a script which loops through each row of the portal may be needed to force a complete refresh. Since this added complexity is onerous and unnecessary for longer portals, an efficient 'Refresh' script for all cases would run along the lines of:

- ‡ If ["Count(PortalRelationship::KeyField) > 18"]
- ‡ Go To Portal Row [Select, Last]
- ‡ Exit Record/Request
- ‡ Else
- ‡ Go To Portal Row [Select, First]
- ‡ Loop
- ‡ Go To Portal Row [Select, Exit After Last, Next]
- ‡ End Loop
- ‡ EndIf
- ‡ Refresh Window [Bring to Front]

where 18 is the number of rows defined for the portal on the current layout, PortalRelationship is the name of the relationship on which the portal is based, and KeyField is the name of the field to which the relationship points in the related file.

Saving user-defined dynamic portal sorts

Adventurous users of the more complex dynamic portal sorting features described in the preceding two posts might want to consider adding the option for users to save multi-key sort configurations and reinstate them from a pop-up menu of 'custom sort' options.

The steps required to achieve this are:

1. Create a related file ('savedsorts.fp5') which holds a 'SortName' field plus the required set of key and reverse-sort fields to define a sort (as outlined in the previous sections of this paper)
 2. Create a 'gSortName' field defined as the key-field in a relationship to 'SortName' field in the savedsorts file. The relationship should be defined to "allow creation of related records" and the gSortName field should be placed on the layout of the main file as a pop-up menu field (value list based on the
-

'SortName' field in 'savedsorts', and 'Include "Other.."' item to allow entry of other values' enabled).

3. A calculating field in the main file should be created to mirror the gSortName field, and defined as the key field in a relationship from the main file direct to the savedsorts file.
4. The key and reverse sort fields from the savedsorts file should be placed directly on the layout of the main file (above the portal) and attached to a value list of fields to be included in the sort (as described previously for the global sort and reverse-sort fields).

Now each sort can be given a name by the users (when they select 'Other...' from the pop-up menu) and the characteristics they define for the sort will automatically be saved in the savedsorts file. Once created, a sort will be available to all users (in multi-user mode) for selection from the menu based on the gSortName field.

Prepared by:
R J Cologon, PhD
NightWing Enterprises
Melbourne, Australia

CobaltSky@nightwing.com.au
<http://www.nightwing.com.au/FileMaker/>

© 2002 NightWing Enterprises

* FileMaker Pro is a Trademark of FileMaker Inc.
