

FindIt

by Mary McNabb, NeXT Developer Support Team

Overview

Welcome to the FindIt Mini-Example!!

The FindIt example, besides displaying some truly fine quotes, implements a basic find panel. The find object can apply a text search to the contents of a text object or a matrix of text fields. The find panel which is implemented in this example is a useful object which can be easily integrated into an application.

Program Organization

How to build the nib files

The nib files in this example are fairly straightforward.

Here are the main points for creating the main nib file:

- Drag a scroll view into the main window, and fill it with text.
- Drag out a textfield, set its attributes, and then control drag it to form a matrix.

You will need a controlling object. Create one, and give it two outlets, one for the scrollview, and one for the matrix. Make sure that when you make the connection to the matrix, that you have the matrix highlighted, and not just one of the textfields within the matrix. You will have to obtain the name of the text object programmatically. You do this by asking the scrollview for its docView.

Here are the main points for creating the find panel nib file:

You will need a text field where the user can type in the string for which they are searching. You will need buttons that actually do the search, one for searching forward, and one for searching backward.

We have included in our example a toggle which indicates whether or not to ignore the case of the letter in the search. We have left it as an exercise for the reader to expand this example such that the search will succeed only if a word matches exactly, or only if it matches an entire word, or only if it matches the prefix of a word.

The find panel is in a separate nib file to save launch time. An application will launch more quickly if it only loads the nib files that it needs at start up. A new window or panel is only created if it is requested by the user. However, since we have put the find panel in a separate nib file, we are faced with the difficulty of not being able to connect the items in the main menu to the find object. So, the Controller object must take a message from these menu items for the find object and pass it on.

User Interface

Note that in our find panel we have followed conventions already set by existing applications. Carriage return causes a forward search, and the command-key equivalents are the same as in other applications. It is important to follow such conventions for the convenience of the user.

Major Classes in the Application

Controller	Serves as an interface object between the user interface, and the underlying objects.
FindObject	The heart of the application. This object implements the routines which control the text search. One of its instance variables is "searchMe". This is the object (either the text object or the matrix) which is to be searched. (Sometimes the searchMe is not correct, however, for the purposes and scope of this example it didn't make sense to bullet-proof it.) Look closely at the findInText , findInMatrix , and searchMatrixStrings methods.

Files

FindObject.[hm]	Implements the find routines. This is the where the core of this example is to
-----------------	--

be found.

Controller.[mh] The object which communicates user requests to all of the other objects.

cUtils.[ch] Some utility routines that implement a brute force search on a text string.

FindPanel.nib The UI for the find panel.

InfoPanel.* The object and user interface for the Info Panel.

magnify.tiff The icon for the application.

FindIt* Generated by Interface Builder.
Makefile
IB.proj

Not tested for 1.0
Valid for 2.0