About **MiscShell**

This object, which combines a MiscSubprocess for process execution and a MiscStringArray to hold the results, is useful in two ways.

· As a quick way to capture the output of a Unix command in a MiscStringArray

```
id who = [[MiscShell alloc] initWithCommand:"who"];
printf("There are %d users\n", [who lineCount] );
printf("The first user is %s\n",
       [[who line:0 field:0] stringValueAndFree] );
```

· As a way to hook up Unix commands and shell scripts to UI objects with Interface Builder.   You can build some interesting little apps this way without writing any Objective-C code at all.   Check out the Samples directory.

Why I Wrote This

Well, for a long time I've wanted to be able to run the Unix **ps** command and display its output in a DBTableView, and to drag the columns around and have it re-sort things.   You can create an app to do that with MiscShell without having to write any Objective-C at all -just a shell script that does "ps aux" and a few connections in Interface Builder.

I got a bit carried away though.

Using MiscShell

You'll find a palette here, **MiscShell.palette**, that contains a MiscShell object (plus a couple of other objects mentioned below), and a library **libMiscShell.a** which contains the MiscShell code.   You'll need to link libMiscShell.a with your application, so you'll probably want to put in in /usr/local/lib or wherever.

Loading the palette is a bit problematic.
The MiscShell palette uses a MiscString internally, so you must have already loaded the **MiscString.palette** that comes with MiscKit. Then you should be able to load MiscShell.palette.

Also, the MiscShell palette contains the code for MiscStringArray and MiscSubprocess, so if you have another palette that needs these objects, it might be tricky to get all of them to load together.

The libMiscShell.a library contains a few extensions and fixes to some of the MiscKit objects, which may be removed in a future release if the fixes make it into the main distribution. Specifically, it adds

- read: and write: methods to MiscStringArray
- a subclass of MiscSubprocess called **MyMiscSubprocess**, which contains
  - a **free** method that calls _childDidExit,
  - €an - **init:withDelegate:keepEnvironment:withPtys:** and **execChild:**

methods that let you pick a different interpreter than /bin/sh (currently unused by MiscShell though)

· a **consumeAllOutput** method that reads all the output available from the subprocess and returns when it's done.

Any and all comments and suggestions on this palette are welcome.

Steve Hayman
shayman@Objectario.com
August 23 1994

Special Bonus Undocumented Palette Objects

There are a couple of other objects on the palette that I find useful.   They don't have much to do with the MiscShell object, but I needed to put them somewhere.

MiscAwakeAction.tiff ¬ This is a **MiscAwakeAction**.   It's a simple little object - all it does is send the target/action message of your choice after the palette is loaded (or after you go into test mode in Interface Builder.)   I often find myself whipping up little demo apps that should do something like "fetch all records" the moment the application starts, and this object saves from typing, oh, maybe 2 lines of code.   (Plus it works in IB test mode. Handy for testing your MiscShell scripts - have it send **executeScript:** to the MiscShell.)

The icon is supposed to be a bugle sending a message to an object telling it to wake up like maybe you might get in army boot camp or something, and ok it's kind of lame but I like making icons by cutting pictures out of Webster.app

The other one that looks like a scrolling Text object is an **EmacsText**, a subclass of Text that implements the Emacs key bindings (inside a ScrollView).   From Julie Zelenski's miniexample.   Currently I don't think it's initializing itself quite right - it's coming up with the wrong default font.   You can set the font yourself if you like in its inspector to get around that one.