PowerModules for PowerBar
A Development Guide
by Scott A. Johnson

Introduction

PowerModules are an easy way for developers to extend the functionality of PowerBar. Using the very simple programming interface, you can perform a variety of tasks without the overhead of an application. This might be useful for "quick & dirty" tasks that the user might perform once in a great while, but not enough to warrent a full blown application.
The possibilities really are endless!

Using the PowerModule "API"

Writing a PowerModule is extraordinairily easy! You need only keep a few things in mind. If you have ever written a CDEF or MDEF, then this will be like child's play :-)

• The PowerModule File (the one the user will drag to an empty button)
You only need to be sure that this file is of type 'PMod' whose creator is 'PBin'. All your resources should reside within the resource fork of this file. You should create this file first and create the resources you need. Note particularly that, although not required, a general "About..." alert or dialog should be included. Do not give this file a 'BNDL' resource.
You can create the file in ResEdit (if you choose) and then later change the type and creator of the document to the above.

• The PowerModule Code Resource
This is a code resource that also resides in the resource fork of the newly created PowerModule file (described above). Please note the following attributes that this code resource must have:

o It must have a type of 'PMod' (set in THINK C's Set Project Type… dialog)
o It must have the following "load" attributes: rSysHeap + rPurgeable
o It must be less than 32K (multi-segment code resources are NOT supported)
o It must have a main entry point defined per the following prototype:

```
pascal OSErr main (
  short                message,      // what do do
  ProcessSerialNumber  *finderPSN,   // Finder's process
  Boolean              *wantFiles,   // do we want to use files?
  Boolean              *wantAliases, // do we want aliases if available
  short                numFiles,     // how many files to process
  FSSpecArrayPtr       files         // the files
);
```

The module will do its work by responding to messages that it receives from PowerBar, it's "parent" process. Note the the System Heap is made current before a PowerModule is executed, therefore, any allocations made (via NewPtr or NewHandle) are allocated in the system heap. This may change in the future.

There are only 4 messages that a PowerModule will receive from PowerBar:

kInitModule(0) - This message is sent upon "first click". It is here where the PowerModule must set the wantFiles and wantAliases flags to indicate if it wants to receive the selected files and, if so, whether it wants PowerBar to resolve any aliases or just send them directly to the PowerModule. It is here, too, that the PowerModule should do any testing of the system environment (like seeing if the Speech Manager is present if it uses it).

kPerformAction(1) - This message is what actually causes the PowerModule to perform whatever action it was created to perform.

kDoAbout(2) - This message will be sent when the user presses the COMMAND key whilst clicking the PowerModule's toolbar button. You should do something nice like put up an "About..." alert or something.

kNoFiles(3) - This message will be sent when the user clicks the PowerModule's button but does not have any files selected when the PM explicitly indicated that it needs files. Again, an alert might be appropriate here.

That's it! See the enclosed sample for the general idea of how things are put together.

A Plea for Help

This feature of PowerBar is in its infancy. I do not claim to have the whole process worked out to perfection. It seems to work quite nicely for very simple tasks, but there may be other factors and nuances that I, in my unenlightened state, may have overlooked.

Therefore I humbly beg of anyone with deeper knowledge than I about such topics contact me with suggestions, ideas, curses or whatever you want. Any help is welcome!!

Farewell!

Have fun! If you write a really neatsie-keeno module, please send it to me!! Perhaps it will be included in a future release of PowerBar (with your permission, of course).

Best Regards,
Scott A. Johnson
October 3, 1993

AOL: DevScott
AppleLink: johnsos
CIS: 71035,3273
Internet: devscott@aol.com