BackMenu V2.20

BackMenu is a program that allows you to define a pop-up menu on the Windows 3.0 background and use it to quickly run applications and tools. The idea came from using X-Windows and Sunview, where the user has a root menu that can be configured.

It allows you to describe a set of actions and associate command lines with them. To bring up the menu simply click with the mouse button (left, middle or right) on the Windows desktop. This will bring up the description list. Selecting a description will cause the appropriate command line to be executed.

Installation instructions are after the next section (just want to tell existing users what's new).

What's New!

* The BackMenu menu file can now have comments in. Any line starting with a semi-colon (;) will be ignored.

* A new keyword, $Info has been added. This tells you (hopefully) useful information about your system (namely what mode Windows is running in and how much free memory is available).

* A browse feature has been added to the execute dialog box. This allows you to wander the disk in search of that file you really want to run. The file types that are listed here can be configured from the options dialog box.

* BackMenu can now auto-run selected items in the menu file when it is first invoked. See the section on making BackMenu auto-run applications.

* BackMenu is now Shareware. We don't like to be bugged for money by the programs we use, so we won't bug you for money in our stuff, but registering will make Ian very happy. Go on, open ORDERNOW.WRI and do it!

Installation

To install BackMenu, extract the files **BACKMENU.EXE, BACKMENU.INI** and **BACKMLIB.DLL** into any directory named in your DOS path. We use C:\WINDOWS, but you may want to do something else. If you've already got a version of BackMenu, don't throw away the old .INI file; you can still use it with this version.

Now from the Run... dialog of Program or File Manager, type BACKMENU. A pretty box will appear in the middle of the screen along with a picture of a watch with the second hand ticking. After that, absolutely nothing!

Now move the mouse pointer to somewhere on the desktop background (i.e. not in any application window) and click the RIGHT mouse button. A menu will pop up on your screen. Select an application from the menu (or perhaps the About... option?). OK, that's all you need to know. However, you'll probably want to alter the contents of the menu to suit yourself. All the information on the menu comes from the Configuration File, which is an ordinary text file you can edit with NOTEPAD.

Configuration File

By default, BackMenu looks for a file called BACKMENU.INI in the same directory as BACKMENU.EXE. The menu file name can be changed by using the 'Set Options...' option.

A configuration file is simply a set of lines (one for each description/command line pair). Additionally, comments may be placed within the menu file. These are lines that start with a semi-colon (;). Any text following it on that line is ignored.
The syntax for a menu item is:-

```
      Menu Item description,application,optional start-up dir
```

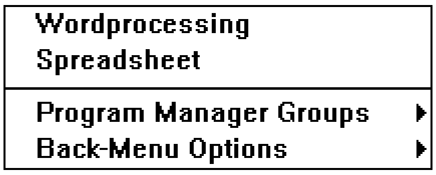
Blank lines may be placed between items, to cause a separator line to be placed in the menu.

Example:

```
      ;
      ;Example BackMenu menu file
      ;
      Wordprocessing, Write,C:\Work
      Spreadsheet,C:\Windows\Excel\Excel

      Program Manager Groups, $Groups
      >BackMenu Options
            About...,$About
            Edit Menu,Notepad.Exe c:\windows\backmenu.ini

            Exit Windows, $ExitWindows NoConfirm
      !
```



BACKMENU also supports cascading menus. A cascading menu item is one that has multiple options associated with it. By clicking on the item, another menu appears with the options on it. It is even possible to have further cascading menus associated with one or more items within that menu.

Associating multiple items with a single menu item is simple. The definition is:-

```
>Item Name
      Sub menu item 1,program
      Sub menu item 2,program
      ...
!
```

That is, the sub-menu is enclosed between a '>' and a '!'. The text that follows the '>' is the text to appear in the menu (a description of the sub-menu).

In the example, 'Name' will appear in the menu, and when clicked upon produce a sub-menu with the items defined. Defining items within a sub-menu is done in exactly the same way as for a main menu (even to the extent of adding a sub-menu to this sub-menu). Eg:-

```
>Editors
```

```
      Andy's editor,AE.EXE
      Notepad,NOTEPAD.EXE
      >Word processors
            Word for Windows,WORD.EXE
            Windows Write,WRITE.EXE
      !
      Multi-Pad,MULTIPAD.EXE
!
...
```

There is no limit to the number of items you can have in any particular menu.

Start-Up Directory

BackMenu will change to the directory containing the executable program and then run it if a path is specified for the executable; thus :-

```
      Corel Draw,C:\CORELDRW\CORELDRW.EXE
```

will change to the directory C:\CORELDRW and then run CORELDRW.EXE. If no path is specified, the application will be started in the current directory (whatever that happens to be).

It is possible with BackMenu to start up an application and set a *different* working directory for that program. To do this, specify the optional start up directory after the application. Eg. excel lives in c:\excel. We want to have excel start up in a work directory d:\data\excel. To do this, the menu option would be:-

```
Excel + Work directory,c:\excel\excel.exe,d:\data\excel
```

Special BackMenu Keywords

All the menu options in BackMenu are configurable.

To fiddle with them, you specify a menu string, followed by a comma followed by a KEYWORD which denotes which BackMenu function you wish to associate with the string. All keywords start with a dollar ($) sign; the case of the keywords is unimportant. Certain keywords may have options following them that will somehow modify their behaviour, these are shown in square brackets **[ ]** - don't type the brackets in the .INI file. A list of the available keywords follows:-

$About Gives version and shareware information about BackMenu. It will be added to the bottom of the menu if it is not specified elsewhere.

$Execute Brings up a dialog box that allows command lines to be typed. BackMenu remembers the last 10 commands typed, and displays the previous command for you to edit. To recall the last 10 commands, press the button at the end of the command line. Click on the command you want to repeat and it will be copied into the command line. There is also a browse button which, when pressed, will present you with a dialog box listing all the .COM, .EXE, .BAT & .PIF files in the current directory. Using this dialog box, you are free to roam the disk in search of the program you wish to run. Clicking on the Open button will cause BackMenu to execute the program displayed in the filename box.

$ExitWindows [NoConfirm]

Exit windows and return to DOS. This option will be added to the bottom of the menu if you do not specify it elsewhere. BackMenu will ask you to confirm before leaving Windows. If you specify **NoConfirm** however, it will not ask, and return to DOS immediate.

$Groups Creates a sub-menu containing the program manager groups. This is described below.

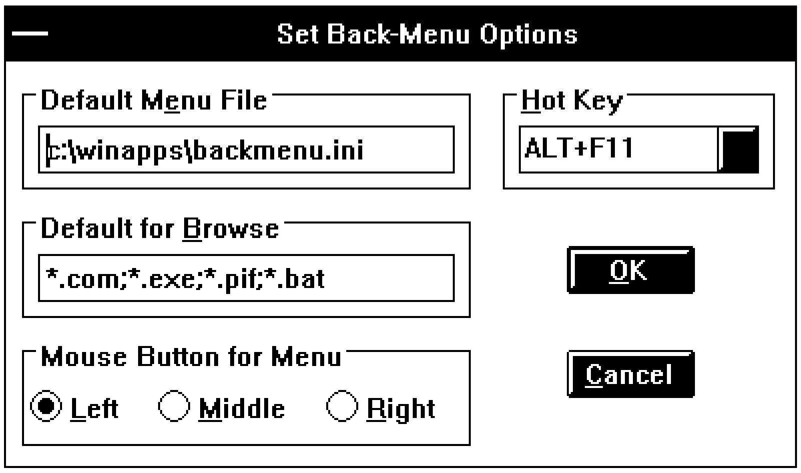
$ReloadMenu [NoGroups]

When selected, causes BackMenu to re-load it's .ini file. Normally, reloading the menu will cause BackMenu to re-scan the program manager groups (if you have the $Groups keyword in your menu). However, if you specify **NoGroups** as an option to this keyword, BackMenu will NOT do this, but use the information from the last time it read the groups. This will make reloading the menu faster.

$RemoveMenu [NoConfirm]

When selected, this option will remove BackMenu from Windows. Normally, BackMenu will ask you to confirm that you really want to do this. If you specify **NoConfirm** as an option, BackMenu will not ask before removing itself. This option will NOT be available if you are using BackMenu as the shell even if you include it in your .INI file.

$SetOptions When selected, brings up the options dialog box to set .ini file and mouse button. The dialog looks like this:


Set Back-Menu Options

Default Menu File: c:\winapps\backmenu.ini

Hot Key: ALT+F11

Default for Browse: *.com;*.exe;*.pif;*.bat

Mouse Button for Menu: [✓ Left] Middle Right

The options here are as follows:

Default Menu File
If you would like a file other than BACKMENU.INI to be used as the initialisation file, you can name it here. It will be loaded straight away and will be used for initialisation next time you restart BackMenu.

Default for Browse
With this option, you can select which files are presented to you

when you use the browse option from the execute dialog box. You are allowed any set of ambiguous filenames, separated by semi-colons. The default is to show all the .COM, .EXE. .PIF and .BAT files.

*Mouse Button For Menu*
Fairly obvious; see the discussion elsewhere.

*Hot Key*
From anywhere in Windows (famous last words!) except a DOS box, you can press Alt+a key to make the BackMenu pop up. By default this is set to Alt+F10, but you can change it by pressing the button next to the description and picking a new key from the list.

$CallDLL *DLLName FunctionName Parameters*
Causes the named DLL to be loaded and the function within that DLL to be called. Anything following the function name are passed to the function as a string. For more details, see the section on extending BackMenu.

$Info Causes a dialog box to appear containing information about the state of Windows. Currently, this contains the mode that Windows is running in and the amount of free memory available.

Example of using keywords:

```
About...,$About
Remove BackMenu,$RemoveMenu
Exit Windows, $ExitWindows NoConfirm
```

Using BackMenu as the Start-Up Shell

BackMenu can be used in place of Program Manager as the default shell for Windows. To install it simply edit SYSTEM.INI which is in the windows directory. In the [boot] section of the file there is a line which will read:-
```
shell=progman.exe
```
Replace it with:-
```
shell=backmenu.exe
```
and you're away. We've noticed that some ill-behaved installation routines set this back without asking; sorry, we can't alter other people's inconsiderate coding! Similarly, some install routines expect Program Manager to be running; just start a copy before starting the installation.

Choice of Mouse Button

The choice of which mouse button to use dramatically changes the way BackMenu "feels". If the left mouse button is chosen, the menu pops-up when the button is pressed and disappears as soon as the button is released. With the right mouse button the menu remains on screen until a bottom-level selection is made, regardless of how many times the right button is pressed.

BackMenu also supports the middle mouse button. If you choose this and find that BackMenu no longer functions then take the following steps:-

1) Exit Windows (or reset if BackMenu is installed as the shell).
2) Edit WIN.INI in the Windows directory.
3) Find the line which has [BackMenu] on it. Below this there will be a line *Button=Middle*. Change this to *Button=Left* or *Button=Right*.

Ambiguous File-Name Support

Back_menu now allows simple ambiguous file names as command line parameters to programs. If BackMenu finds an ambiguous file name, it will prompt with a dialog box to let you choose. The path and filename are substituted in to the command line. If abort is chosen, the application is not started.
Eg.

```
      Word, C:\WORD\WINWORD.EXE *.doc
```

At the moment, paths cannot be specified, and the command line may only contain the ambiguous file name.

Command Line Parameters

It is possible to make BackMenu prompt for a command line, which is then passed to the relevant program. To do this, place a % after the program name in the menu file. Eg.

```
      Excel,c:\excel\excel.exe %
```

BackMenu will bring up a dialog box, which will allow the command line parameters to be typed. Anything following the % will be placed in the dialog box for editing.

Program Manager Groups

BackMenu has the ability to read program manager group files and provide the items and groups as a pull-right menu. If you create a menu item with the keyword **$Groups**, BackMenu will insert a pull-right menu containing all the program manager groups. Pulling right on a particular group will give a list of the programs within that group.

Invoking BackMenu from Other Applications

The BackMenu DLL (BACKMLIB.DLL) contains an accessible function that will allow the menu to be popped-up remotely. The function is called **ActivateBackMenu()** and takes no parameters. By calling this function it is possible for other applications to cause the background menu to be displayed.

Here is a Microsoft Word-For-Windows macro that will do just that:-

```
Declare Sub ActivateBackMenu Lib "BACKMLIB.DLL"
Sub MAIN
      ActivateBackMenu
End Sub
```

Writing a similar Excel macro is left as an exercise for the reader.

A note about WINCOM.DLL

WINCOM.DLL is a DLL that provides standard dialog box functions for Windows applications. If you already have WINCOM.DLL from another application, then BackMenu will quite happily use that version.

Extending BackMenu

The **$CallDLL** is a powerful hook, through which the functionality of BackMenu can be extended. Programmers may write functions within a DLL that performs some function, and have it executed through BackMenu. Parameters may be passed to the function in the form of a string as this allows the most flexible way to pass parameters of different type. The % and * characters may be used to prompt for a parameter or get a file name from a list as with the other keywords. A prototype for a C function that could be called from BackMenu is:-

```
/
*-------------------------------------------------------------------------------------------------------------------------
*/
void  FAR PASCAL  ShowBox(LPSTR Params)

{
    MessageBox(GetActiveWindow(),Params,"BackMenu DLL Test", MB_OK);
}
/
*-------------------------------------------------------------------------------------------------------------------------
*/
```

A Turbo Pascal program that would work is as follows:

```
LIBRARY CommandLine;

{Sample procedure callable using $CALLDLL from BackMenu.}
{Not even slightly copyright.}

USES
      Strings, WinProcs, WinTypes;

PROCEDURE ShowBox(Params : pChar); EXPORT;
BEGIN

    MessageBox(GetActiveWindow, Params, 'BackMenu DLL Test', mb_Ok)

END; {PROCEDURE CommandLine}

EXPORTS
      ShowBox ;

BEGIN

END.
```

If you write any useful DLLs which use this facility, please let us know.

Making BackMenu Auto-Run Applications

It is possible to mark applications within the menu so that, when BackMenu is invoked, the marked applications are run automatically. To do this, insert an at symbol (@) before the command line for that item. Eg.

```
;
;Example of an meny using AutoRun feature
;
>My Apps
	Clock,@clock.exe
	Screen Saver,@c:\winapps\saver\Saver.exe

	Free Memory,FREEMEM.EXE
!
```

In the example above, clock.exe and saver.exe will be executed when BackMenu is run. If you use the choose to reload the menu file, however, the @ will be ignored. This allows you to use BackMenu to decide which program you want running.

There may be times when you've selected some applications to run, but don't actually want them to start up when you run BackMenu. You can do this by holding down either shift key while BackMenu is loading. This will cause BackMenu to ignore any @'s that you've placed in the menu file.

Hope you like BackMenu. It's become indispensible for us. We welcome suggestions and comments by electronic mail to **ih@ecs.soton.ac.uk** or to **sphipps@cix.compulink.co.uk**; as ever, these addresses will need reversing, munging and otherwise manipulating to make them work on the network. We will always send a reply to your note; if you don't get one, send it again trying an alternative (reverse the bit after the @; drop the cix bit; add a gateway name to the end; ask your local mail guru; etc). No promises of support or implementation but we're all fairly nice people (or so our mothers say).

Postal contact should be to:

SP Services
PO Box 456
Southampton
United Kingdom
SO9 7XG

Sorry, we don't take phone calls (apart from registrations on +44 703 550037!)