# BackMenu V2.34

## Contents



## 1. Introduction

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.

BackMenu is shareware. That means you are welcome to use it as much as you like to evaluate it, but as soon as you like it & decide to make it part of the way you work, you must pay for it! Please see the accompanying Write file, ORDERNOW.WRI, for more details. BackMenu is a part of the BackDesk package, but does not depend on other parts of the package.

## 2. Installation

To install BackMenu, extract the files **BACKMENU.EXE**, **BACKMENU.INI**, **BACKMLIB.DLL** and **PMGROUPS.DLL** into any directory named in your DOS path. We use C:\WINDOWS, but you may want to do something else. If you're running Windows 3.0 you'll also need to extract **COMMDLG.DLL** and place this along with the other BackMenu files. Windows 3.1 owners already have this DLL as a part of the Windows installation. 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, but you can now throw away **WINCOM.DLL** as long as nothing else needs it.

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 (or perhaps some juggling balls). 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.

## 3 Using BackMenu

Now you've installed BackMenu, you'll want to create your own menu for the program (unless by some strange quirk of fate, we've managed to describe your entire setup in our example menu file).

### 3.1 The 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 given below. Items contained within square brackets are optional.

```
Menu Text,[@][+|-]application [%]parameters [,start-up dir]
```

or

```
Menu Text,$Keyword [Keyword Parameters]
```

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

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).

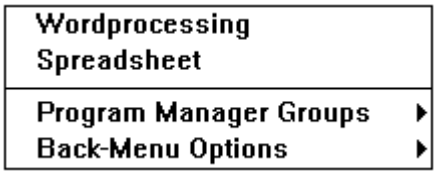
Example:

```
;
;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
      !
```




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.

Right, here's all the different things you can specify as a BackMenu menu item.

### 3.2 @ - 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.

### 3.3 + and -  - Starting Up An Application Maximised or Minimised

There may be certain programs that you want to start up in certain ways. Whilst BackMenu doesn't let you position the application (at least, not yet), you can say whether you want it to startup as an icon (minimised) or full screen (maximised). To do this place a minus or plus before the application name. Eg.

```
;
; Example of using + and -
;
>Network software
      ; Start the WinQVT/Net server up minimized
      Gateway to the world,-c:\winapps\wnqvtnet.exe
      ; Start X-Vision up as full-screen
      XVision,+XVISION.EXE
!
```

You can use this option with the auto-run feature for even more fun. Eg.

```
      Gateway,@-c:\winapps\wnqvtnet.exe
```

will cause the WinQVT/Net server to be started up when BackMenu is initially loaded *and* cause it to be iconised.

A WORD OF WARNING - + and - override the defaults for any starting program. It is possible to get certain applications very confused by starting them up in a mode they're not suited to.  Be careful using them.

### 3.4 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 and before and parameters 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.

### 3.5 Ambiguous File-Name Support

BackMenu 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.

### 3.6 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
```

### 3.7 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 your program manager groups. Activating this menu lists the groups and activating a group displays the programs available in that

group.

**$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:

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.

*Editor For Menu File*
Every time BackMenu finds that the menu file has changed, it will be reloaded. If an error is encountered during the auto-load, the program whose name is given here will be invoked to allow the error to be corrected. By default (i.e. when this field is blank) the command:

NOTEPAD *menu-file-name*

is given; you may use any editor which will accept the name of a file as a command line parameter.

*Automatically Reload Groups*
If you check this box, the Program Manager group files will also be automatically reloaded when the menu file changes (although not when the .GRP files change). You'll probably prefer to leave this option unchecked (the default).

*Run DOS Command Line Server*
(386-Enhanced users only!) This enables or disables the BackMenu command line server that allows you to run Windows programs from a command shell within Windows. For more details, see the section on the DOS Command Line Server.

**$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.

**$Tasks**
Provides a pull-right menu that contains within it a list of the Windows applications that are currently running, displaying the title-bar of each. By selecting an item from this menu you can switch to that application.

Example of using keywords:

```
About...,$About
I'm running,$Tasks
Remove BackMenu,$RemoveMenu
Exit Windows, $ExitWindows NoConfirm
```

# 4 Advanced Features

## 4.1 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.

### 4.2 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*.

Note that the third mouse button is not supported by many mouse drivers; if you find it doesn't work with BackMenu, please ensure that you have a driver that supports three button operation under Windows.

### 4.3 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.

### 4.4 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.

### 4.5 The DOS Command Line Server

Have you ever been at the DOS prompt in a Windows session and thought 'I'd like to type TOWERS.EXE as a command and have it start up within Windows.'? Well, if you're the proud owner of a 386 or 486 machine and are running windows in Enhanced mode, then this can be a reality. BackMenu now comes with a DOS utility, WRUN.EXE which lets you get run Windows applications from the DOS prompt. Simply type:

```
      WRUN Command line
```

and (if all is well, and you've not disabled the server option), the program you typed as part of the command line will be run in Windows. You can type any of the BackMenu optional thingys as part of the command line. Eg.

```
      C:\> WRUN -C:\WINAPPS\WNQVTNET.EXE
```

to run the WinQVT/Net server as an icon, or

```
      C:\GAMES> WRUN EXCEL.EXE,C:\WORK
```

to make Excel start up with C:\WORK as the default directory. WRUN will also start up Windows if it isn't already running, and pass your command line on (This featre works on ALL versions of Windows).

And there's more!! WINSTART.EXE is a windows stub that developers can glue to their applications so that they will start up automatically if typed from a command shell running under Windows, or start up Windows and pass the program name otherwise. For example, BigDesk has this stub attached and so you can type:

```
C:\>BIGDESK
```

from a Windows shell and it will run!!! This will also work with our very own Arachnid, Towers and Mines (try it!)

Disabling WRUN & WSTART

WRUN and WINSTART use the clipboard to communicate with BackMenu to pass the command to be executed. This means that any text on the clipboard will be trashed whenever you use them. This may be an undesirable feature so, like all good features, it can be disabled. Both WRUN and WINSTART look for an environment variable called NOWINSTART and, if it exists, return an error rather than using the clipboard. Thus:

```
C:\>set nowinstart=TRUE
```

will disable WRUN and WINSTART (try it).

```
C:\>SET NOWINSTART=
```

removes the environment variable and so WRUN and WINSTART will work again.

WRUN and WINSTART are both freeware - you may distribute them to anyone and include them in any other packages without paying any royalties. However, we would like you to tell us if you use WINSTART in a program which gets distributed so that we can include your program in our list of compatible software (free advertising!) If you would like a version of WINSTART customised in any way (e.g. remove our little plug, or add your own), we can supply this for a small fee - please contact us at the address shown below.

### 4.6 Auto Monitoring of the Menu File and Menu File Parsing

V2.34 of BackMenu provides two new features. Firstly, BackMenu will now parse the menu file before attempting to create your menu. If BackMenu encounters an error, you have the option to load an editor and edit the menu file, and the currently loaded menu will be left unchanged. BackMenu will insert a comment below the line which caused the error (or at the end of the file if the error is a missing "!"). The editor used is configurable and can be set from the $Options dialog box. If no editor is specified (ie. the entry is blank), BackMenu will default to using Notepad. Currently, BackMenu checks for lines that are too long, contain invalid keywords or problems in the definition of cascading menus.

Secondly, BackMenu will automatically monitor the menu file and reload it if the file is modified. It checks the file every time the menu is accessed and a small delay is encountered if the menu has to be reloaded (along with a message and the usual animated cursor). The menu is parsed at this time as well and so it is

possible for an error dialog box to appear as described above.

## 5 What is (& was) New!

### V2.40

* BackMenu now understands more of the 3.1 PM groups (at least it now understands the minimized flag).

### V2.34

* Fixed Windows 3.1 group problems, the system now decodes 3.1 groups properly. Also the group file reading has been speeded up (it's now as fast as we can imagine it going!).

* Changed from using WINCOM.DLL to COMMDLG.DLL. Existing users may delete WINCOM.DLL as long as no other applications are using it...

* Made browse... option from execute remember last directory used.

* BackMenu now decodes Program Manager's startup group if it is the shell and executes the programs it finds there.

* Added $TASKS keyword that provides a pull-right menu of the executing tasks.

* Tidied up dialog boxes so they work with the keyboard.

* Fixed a bug in the clipboard monitoring that caused a crash if large amounts of text were cut or copied to the clipboard.

* BackMenu now monitors the menu file. If the file is modified, BackMenu will automatically reload the file the next time you try and access the menu.

* BackMenu now parses the menu file before creating the menu. If an error is encountered, you have the option of invoking an editor (the default is Notepad) to correct the problem. BackMenu will insert a comment into the menu file showing the location of the error.

* The menu file loading has been speeded up (again).

* An alternate animated cursor was added (juggling balls). Other ideas welcome...

### V2.33

* Internal revision.

### V2.32

* Several cosmetic and speed enhancements.

### V2.30

* You can run Windows programs from the DOS prompt. For more details, see the section on the DOS Command Line Server.

* You can start applications up iconised and maximised. By placing - or + in front of the command line, you can tell BackMenu how you want the

program to start up.

**V2.21**

* BackMenu is now BigDesk aware. If you pop-up a menu and it overlaps the BigDesk window, it will not be obscured if you have BigDesk's keep-to-front option enabled. This requires BigDesk V2.30 or later.

**V2.20**

* 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.

**V2.00**

* 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!

## 6 Thank you for reading this document

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!)