Help with dwm

 Dwm, the **desktop window manager**, is one of a suite of dwm applications which work in concert, giving the user control over the desktop environment, Windows programs, file launching and file manipulation.
Dwm is designed to be used as a Windows shell. It's main interactive function is to act as a **program manager**, and provides an interface for quick launching of applications through pop up menus. Items can be launched directly from a pop up menu, or dragged off the menu and dropped onto the desktop, leaving a **blanch** (button launch). Dwm is also a **desktop manager,** keeping track of all the desktop objects between Windows sessions (this includes the states and positions of vern, dobs, blanches and the dwm button). Dwm also is the inter application communication center for the dwm applications, and must be running for vern, blanch and dobs to work.
To learn more about dwm and its abilities, click on one of the following topics. For help on one of the other dwm applications, click on its button on the help window button bar. If you need help on how to use help, press the **F1** key.

Running dwm

Once you properly install (which you have done, otherwise you would not be reading this help file), dwm is available for execution the same way any other Windows program is run (i.e.: by **double clicking** on the dwm icon in the dwm group of the Windows Program Manager). If you have chosen to use dwm as the **shell**, it runs automatically whenever Windows is started. When dwm is run, it automatically enables **blanch** (the button launch facility), and may also run **vern**, the virtual environment manager (this is an option which is set in the **dwm options dialog box**). It may also restore desktop objects generated by other applications in the dwm application package.

Initially, the only visible manifestation of dwm is the dwm button, which appears on the screen as a dark gray square containing a series of smaller nested squares. Double clicking on the button (or right clicking), or hitting **Alt+Enter** when the dwm button has focus opens the **dwm options dialog box**. It is through this dialog box that environmental variables are set, desktops selected, loaded or saved, and menus assigned to left and right mouse clicks.

Clicking on the desktop background (any area on the screen not covered by a program window or a desktop object) reveals a **pop up menu**. When dwm is first installed, a **left click** generates a menu containing some native dwm functions, along with a few stock Windows applications. Clicking on the desktop with the **right** mouse button pops up a **Program Manager menu**, which is a list of submenus (program groups); clicking on a program group item spawns a flyout menu, listing all the programs contained in that group.

Pop up menus are designed to be user configurable. After revealing a pop up menu, right clicking on the menu calls up the **menu editing dialog box**. Menus can be easily modified and saved or saved as a new menu file. In this way, the user may generate any number of unique pop up menu structures, although only two can be in use at any one time.

How it works...

When dwm is run, it reads the file **dwm.ini**, **desktops.ini** and **blanch.ini** (all located in the WINDOWS directory) to find the desktop file and the menu files to be used, the startup group (programs to be launched immediately when Windows starts) as well as the environmental options for itself as well as the other dwm applications. Note that the file dwm.ini, along with blanch.ini and desktops.ini are the only dwm files that are not written to the dwm directory by default, but are written instead to the WINDOWS directory. They are not present after installation, but are created after dwm is run for the first time.

The desktop file (these files have the extension **.dsk**, and are located in the **DWM** directory) contains instructions for loading blanches and dobs. Dwm refers to the file blanch.ini, to determine the particular characteristics of the blanches appearing on the desktop. These desktop objects will be restored (appear on the desktop) if the *Restore desktop at startup* option is toggled in the dwm options box. The menu files (these files have the extension **.mnu**, also located in the **DWM** directory) contain information for the structures of the pop up menus, which are activated by clicking on the desktop background (any area on the screen not covered by a program window or a desktop object).

If either the left or right mouse click has been assigned to the special pop up menu **progman.mnu**, dwm also scans the file **progman.ini** (located in the WINDOWS directory), as well as progman group files (these have the extension **.grp**, and are also located in the WINDOWS directory) in order to generate the progman menu file. This menu file is initially assigned to the right mouse button by default, though this can easily be changed. Editing the file progman.mnu (i.e.: to rearrange or delete certain groups etc.) will have no lasting effect, as the file is regenerated each time dwm starts up. Users wishing to use a subset (or superset) of the progman group menu should use the **dwm menu editor** to rename the mnu file something else.

The generation of the progman menu takes a few seconds, which translates into a longer startup time. To make the startup somewhat quicker, the user should save the progman.mnu file with a different name, and assign this new menu to one of the mouse buttons.

Using dwm as the shell

In technical terms, a **shell** is a program that interprets a users commands then makes calls to a resident operating system in order to perform various functions, including running other programs. The Windows operating system is itself a secondary shell, running under DOS (a primary shell). Secondary shells, such as Windows, can be started and terminated, but in general there is no way to exit a primary shell, short of turning off the computer.

In the Windows operating system, the term shell has a special meaning; is the first program run (automatically) whenever Windows is started, and is specified by the **shell=** line in the **[boot]** section of the **system.ini** file. Typically the shell will be a program that can be used to launch other programs (something like dwm.exe or progman.exe).

To make dwm the shell, open the dwm dialog box by **right clicking** on the dwm button, and set the checkbox captioned **make dwm the default shell** ON. Click on the **Accept** button, and exit Windows.

You should now check that the **PATH** statement in the **autoexec.bat** file contains the directory which includes the dwm distribution (which is c:\dwm by default). The path statement may have already been modified during installation of dwm; if not, edit the autoexec.bat file and reboot the computer. The next time you run Windows, dwm will be the shell.

Note: The statement oldshell= in the system.ini file preserves the line identifying the application used as the shell prior to making dwm the new (default) shell.

The pop up menus

Clicking on the desktop background with either the **left** or **right** mouse buttons reveals a pop up menu. Alternatively, the user can hit **Ctrl+l** or **Ctrl+r** (for the left and right menus, respectively), when the dwm button has focus. The menus are intended to be different, depending on whether they were called with a left or right mouse click. Pop up menus may be a single list of system functions and/or executable programs, or may contain any number of flyout sub menus (each with their own list of functions, program items and sub menus); the structure of the menu is entirely up to the individual user.

Items in the pop up menu can be launched by clicking with the left mouse button (if you click left and hold for a moment and then release the mouse button, the program or function will not launch). If you click and hold an *executable* menu item, it can be **dragged** off the menu and dropped onto the desktop, leaving a **blanch** (button launch) for that item. There is also a special case of an executable item that generates a **Run...** dialog box from which programs can be launched. This Run dialog box retains a memory of the last twenty files or applications launched using it.

The user can move a menu around on the screen **dragging** it by menu title bar to new location. Sub menus can be moved independently of their parent menu in the same way. Launching a program or calling some function from a menu or sub menu will close the pop up menu windows. Similarly, clicking on the desktop background will close any open pop up menus.

To quickly open the *Windows Task Manager* hold down on the **Ctrl** key while left clicking on the desktop. Similarly, hold down **Ctrl** key while right clicking dwm to open the dwm **Options** dialog box.

Note: Information for structure of the pop up menus are contained in the files with the extension **.mnu**, usually located in the directory containing the rest of the dwm distribution. See **editing menus with the dwm menu editor** for details on how to customize your pop up menus.

Pop up Program Manager group menu

When dwm is first installed, clicking on the desktop background with the **right** mouse button will reveal a menu of program groups controlled by the Windows Program Manager. The pop up program manager menu is a list of sub menus; **left clicking** on a group name (or hitting the right cursor key when a submenu item is highlighted) will spawn a fly out menu which lists the applications contained in that program group. Program groups are often manipulated so as to contain related applications (i.e.: graphics tools, communication software, games, etc.), or may be automatically created during the installation of some software packages.

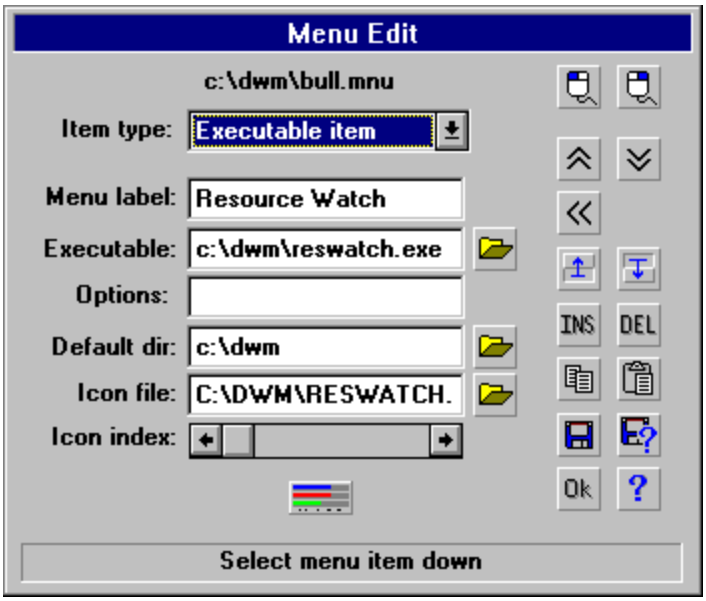
Programs listed on the fly out menus are identifiable by name and include the program icon, if it is retrievable (if not, dwm will supply a default icon). Clicking with the left mouse button on an application will execute the program (if you click left and hold for a moment and then release the mouse button, the program will not launch). If you click and hold an executable program item, it can be **dragged** off the menu and dropped onto the desktop, leaving a **blanch** (button launch) for that program.

Clicking on the title bar of the program manager menu opens the Windows Program Manager, allowing the user to manipulate the program groups. If the Program Manager Groups have been modified, the user should open the **dwm options** dialog box, and click on the appropriate **Refresh** button, so that the changes are reflected in the pop up program manager menu.

Note: The file **progman.mnu** is used by dwm to generate the pop up program group menu, and is regenerated by dwm when Windows is started or when the menu is refreshed, by reading the Windows **progman.ini** and ***.grp** files. For this reason, this file cannot be edited like a normal menu file, although it uses the same syntax as the user configurable menu file. Users wishing to modify the progman.mnu file so that it generates a subset (or superset) of the Program Manager groups should use the **dwm menu editor** to save the progman menu as a menu file with a different name, after which the new menu can be assigned to a mouse click, and edited with the menu editor.

Editing pop up menus with the dwm menu editor

When a dwm pop up menu is activated (by clicking on the desktop), **right clicking** on the menu will call the dwm menu editing facility. The menu editor is used to add, delete, or move menu items around. Menu items can also be copied to a clipboard, and pasted into another menu, or pasted in somewhere else in the same menu. After a menu has been modified, it can be saved, or saved as a new menu file from within the menu editing environment. In this way, the user can create any number of unique menu structures, although only two can be in use at any one time. The dwm menu editing dialog box is shown here; click on various parts of the dialog box for more information about their functions.



The menu editing dialog box: The menu editing dialog box shows the name of the file being edited at the top (in this case, it is the file **bull.mnu**, located in the **dwm** directory on the **c: drive**). When the dialog box is opened by right clicking on a menu item, it initially opens with the chosen menu item ready for editing. The dialog box contains an Item type drop-down list box, and a field for user supplied information for the particular menu item type being edited (the edit boxes in the information field will change, depending on the type of menu item being considered). Along the right side of the edit box are a series of buttons, which are used for editing purposes.

Mouse button buttons: These two buttons are used to switch between the pop up menus assigned to the left and right mouse clicks, respectively. For instance, if you are editing the left menu, you can click on the right mouse button button, which will close the left mouse button menu, and open the right mouse button menu for editing. These buttons are useful when copying and pasting menu items form one menu to another.

Up and down arrow buttons: These are used to scroll through the list of items that make up the menu. As new items are scrolled through, the information field in the menu edit dialog box changes to show the particulars for the highlighted menu item.

Left and right arrow buttons: These buttons only appear when editing submenus, and are used to descend and ascend through a nested submenu architecture. Here, only the left arrow button is visible, indicating that the menu item is part of a submenu to some other parent menu. Clicking on the left arrow button will close the submenu, returning to the parent menu. If the principal menu has multiple flyout menu items, the submenus must be edited separately, starting at the top level, by recursing into, and then backing out of the nested submenu structure.

Item mover buttons: These buttons are used to change the position of menu items within a given menu or submenu. When clicking on these buttons, the highlighted item will move up or down, displacing the other menu items as it moves. Note that items cannot be moved between different submenus in this way - they must be copied and pasted into the desired location.

Insert and Delete buttons: Clicking on the INS (insert) button will create a new menu item, directly below the insertion point. Dwm automatically gives the new item focus, so it can be edited immediately The new menu item is initially a blank template for an executable item, but this can easily be changed by changing the item type. Clicking on the DEL (delete) button will delete the selected menu item. If the item is a flyout menu, the submenu (along with any nested subordinate submenus) will be recursively deleted. There is a confirmation dialog box that appears if the user attempts to delete a submenu, as a safety precaution.

Copy and Paste buttons: These buttons are used for copying and pasting menu items. They are generally used for duplicating or moving menu items between submenus or different menu files. Click on the copy button (the one on the left) to place a copy of the menu item into the dwm menu editing clipboard. Once an item has been copied into the clipboard, it can be pasted into a new position, by clicking on the paste from clipboard button. Note that the pasted item will replace the item it is pasted over, so the user will usually want to create a new item (by clicking on the INS button), and then click on the paste button.

Save and Save As buttons: These buttons are used to save the modified menus. The button on the left, which looks like a diskette, is the **Save** button; if clicked on it will overwrite the current menu file to incorporate any changes to the menu. The other button, that looks like a diskette with a question mark, is the **Save As** button. Clicking on the save as button opens a dialog box which prompts the user to supply a name for the new menu file. The save as button allows the user to create any number of unique menu structures, although only two can be in use at any one time (one assigned to each of the left and right mouse buttons). Assigning menus to the left and right mouse clicks is done through the **desktops dialog box**, accessed by clicking on the **Edit** button in the **dwm options** dialog box.

Accept and Help buttons: These are the last two buttons in the dwm menu edit dialog box . Clicking on the **Ok** button will close the menu editing dialog box, but the modified menu will not be saved. Changes made to the menu while in the editing environment will be reflected in the pop up menu for the duration of the Windows session, but the changes will be lost when dwm terminates unless the menu is explicitly saved. Users wishing to save menu modifications should click on the save button before the Ok button is chosen, or alternatively click on the Save left/right menu button in the dwm options dialog box, before exiting dwm. Clicking on the **question mark** button launches the on line help system for the dwm applications; in this case the help system starts up in the menu editing topic.

Menu editor infobar The information bar at the bottom of the editor window is used to display current action, usually caused by clicking on an editor button.

Menu item types: Clicking on the down arrow at the right of the item type edit box reveals a drop down list of the available menu item types. When creating new items (by clicking in the **INS** button), the user will often edit the entry (new items created with the INS button are initially a blank template for an executable menu item) by changing the item type from the default to the desired type. Choosing an item type out of the drop down box will change the information field, so that the user need only supply information specific to the item type chosen.

There are seventeen different menu item types. Of these, there are four that are used frequently: **Title**, **Flyout menu**, **Executable item** and **Separator**.

The others are for specific dwm functions, which are generally used once or not at all in a given menu structure. They are: **Desktops**, **Task manager**, **Refresh left menu**, **Refresh right menu**, **Save left menu**, **Save right menu**, **Save desktop**, **Load desktop**, **Clear desktop**, **Restart Windows**, **Exit Windows**, **Exit dwm** and **Options dialog**.

Flyout menu: The Flyout menu item is used to create nested groups (i.e.: submenus, or menus within menus) of menu items. In a pop up menu, items that spawn submenus are identified by an arrow at the right of the menu item cell. Left clicking on a flyout menu item will pop up a submenu, which may contain executable items, special dwm functions or other submenus.

To create a flyout menu, first create a new item, by clicking on the **INS** button (the new item will be an executable item, by default), then choose flyout menu from the drop down item type list box. When a flyout menu is created, it is given a default structure, consisting of a title, followed by a separator. It is then up to the user to recurse into the submenu (by clicking on the right arrow button), change the default submenu title to something appropriate, and load up the submenu with new items.

The user is required to supply a label for the flyout menu, as well as a **menu identifier**. The label is simply the text that appears on the flyout menu cell in the pop up menu. The identifier is used internally to organize menu structures for each submenu into separate sections in the .mnu file. For a given pop up menu structure, each submenu must have a unique identifier. Failure to provide an identifier, or giving two or more submenus within a menu file the same identifier will usually cause problems. The main menu of a pop up menu structure gets the name of the menu file as an identifier, by default.

Title: The menu item **Title** generates a title bar on the pop up menu or submenu. The only information needed for this item is a label, which is the text appearing on the title bar of the pop up menu. The title bar looks the same as any other menu cell, but does not call any program or function when clicked on (with one exception, see below). Clicking on a title bar, and dragging the mouse allows the user to move a menu or submenu around on the screen, independently of any other exposed menu. For this reason, it is a good idea for each submenu in the menu structure to have a title. Note that when flyout menus are created, they automatically acquire a default title, which can be customized by recursing into the submenu, and editing the label of the title.

If the user gives the Title item the label **Program Manager** (written exactly like this, with **P** and **M** capitalized), clicking on the title bar with the left mouse button will call the Windows Program Manager.

Separator: This item places a small gap between other items in the menu list, and used for formatting and aesthetics. You can have as many separators as you want in the menu file.

Executable item: This is perhaps the most important of the item types, and also requires the most work to customize. Executable items in a pop up menu can be launched, with prespecified options, by a left mouse click, and can also be dragged out of a menu and blanched. Executable programs usually have the extension exe (or less commonly, pif, bat or com). Files that have extensions **associated** to executable programs can also be included as executable menu items.

New items created by clicking on the **INS** button will initially be a blank template for an executable item. The information field allows the user to enter a **label**, **executable file**, **options**, **default directory**, **icon file** and **icon index**.

There is also a special case of the executable item that generates a **Run Program** dialog box.

Menu label: The menu label is required for all menu items (except the separator), and is simply the text appearing on the pop up menu cell for the item, and is independent of the item function. If the user does not supply a label for a menu item, it's cell in the pop up menu will be blank (i.e.: the user may not be able to tell what will happen if the menu cell is launched or otherwise activated).

Executable field: The **Executable** edit box is reserved for the name of the executable program. Files that have extensions **associated** with executable programs can also be entered into this field. It may also be necessary to include the drive and path specifiers for the program or associated file (i.e.: c:\stuff\secret\myprog.exe). If the path to the executable in question is included in the **PATH** statement in the **autoexec.bat** file, the name of the executable is all that is necessary. Note that at the right of the edit box is a button that looks like a file folder. Clicking on this button opens a **file browsing** dialog box. Using the file browser, the user can peruse the available drives and directories, looking for a specific file to add to the menu. Selecting a file from the browser (and clicking on the Ok button in the file browser dialog box) automatically loads the chosen file, along with the drive and path information, into the edit box.

Options field: This edit box is used to specify start up options, if any, for the executable program. The entry may contain the name of a file (or files) to be loaded into the executable at launch, or may contain command line switches. The type of options and switches supported is entirely a function of the program being launched; many programs do not require options of any sort.

Most programs that operate on other files (word processors, for example), will allow at least a file to be specified as an option (be sure to include the path to the file, if necessary). Certain programs that are designed with a multiple document interface will allow many files to be loaded simultaneously. Programs of this type can have option lines that include several files, separated by spaces.

If the file specified on the Executable line is a file **associated** to an executable program, the options line can be left blank (unless, of course, the user wishes to include valid command line switches for the executable program).

Some Windows programs do not properly process options from the command line, and require files to be loaded and options set from within the program, after it is launched. In this case the supplied options will appear not to work.

The options field is optional, and can be left blank. If no options are supplied, the executable file will start up in its native state, and it is then up to the user to load files or set program options, using the protocol of the program in question, after it is launched.

Default directory field: The default directory edit box is used to specify a default drive and directory for the executable program. When a program is launched, the default directory is normally the directory containing the executable file. Whenever a program tries to create, access or save a file, it uses the default drive and directory, unless some other drive and directory path are specified, and this is where the default directory field comes into play. If a default directory is not specified, the user will often need to negotiate the file browsing dialog boxes, changing drives and directories until the desired source or destination is found. In specifying a default drive, the file browsing dialog boxes will open up, already pointing to the desired drive and directory.

For example, assume the user has a hypothetical word processor called **Milton** on the **c drive** in the directory **wordapps** (i.e.: c:\wordapps\milton.exe), and all the files that they are working on are in another directory on another drive (for example, d:\textfile\personal*.txt). If a default drive is not provided, the dialog boxes for loading and saving files will open in the c:\wordapps directory, forcing the user to manually switch drives and directories until the files in d:\textfile\personal are listed. By setting the default drive as d:\textfile\personal, the dialog boxes will open up in the desired directory.

To the right of the default directory field is a browse button, shaped like a file folder. Select a drive and path from the browser then click on the Ok button to close the file browser dialog box. The default directory edit box will be automatically loaded with the drive and path information.

Icon file : In the iconfile edit box, the user gives a path to the file containing the desired icon. Icons are resources bound into applications, and are used in Windows as a visual cue to quickly identify programs. Icons are most commonly stored in files ending in .ico, .dll or .exe. In dwm, icons are painted onto the menu cells containing executable items, as a reminder of the program launched by the menu item. If no icon file is supplied, dwm assumes the iconfile to be the same as the file on the Executable line (i.e.: the icon will be the normal icon for that executable program). There is a browse button to the right of the iconfile edit box, that can be used to load a desired file into the edit box.

Note that extracting icons from the specified iconfile takes a little time, which can slow down the appearance of pop up menus. To speed up the menu drawing, it is a good idea to include the drive letter and full path to the iconfile, even if the path to the directory is included in the PATH statement in the autoexec.bat file. Certain users may wish to suppress the icon drawing entirely, in which case the iconfile and icon index become irrelevant. Suppressing icons is an option set in the **dwm options** dialog box.

Icon index: Some programs, and certain icon-package utilities contain multiple icons, that the user can choose from in order to further customize their Windows programs. For example, the Windows Program Manager, progman.exe, contains numerous icons. Just below the iconfile edit box is an icon index scroll bar, with the selected icon appearing below the scroll bar. Clicking the left or right arrows on the scroll bar lets the user peruse the various icons available in the iconfile (often there will be only a single choice).

Icon: The icon appearing on an executable menu item is shown near the bottom of the dialog box. Clicking on the left or right arrow buttons of the **icon index** scrollbar may change the icon, if there is more than one available icon in the specified icon file.

File browser buttons: Certain of the information fields for the executable menu item type require a file, along with a path to the file (that is, if the path to the directory containing the desired file is not already included in the **PATH** statement of the **autoexec.bat** file). Clicking on an **open file button** to the right of an edit window opens up a standard file browsing dialog box. Use the file browser to select a file, then close the browser (by clicking on the browser's **OK** button), and the file, along with the path to the file, will be automatically entered into the appropriate edit window.

The Run Program dialog box

There is a special case of the executable menu item that behaves differently from normal executable items. After creating a new item of executable type (by clicking on the **INS** button in the **dwm menu editor**), the user edits the Menu label (to something like Run... or Run This!), and leaves all the other fields empty. This menu item, essentially a blank executable, will generate a dialog box when clicked on. The dialog box looks like this:

The user can click in the file window, and type in the name of the executable program they want to run (along with the drive and path, if necessary). Files with extensions that are associated to an executable application are also valid entries. Clicking on the File button opens a file browsing dialog box, in which the user can look for the file that they wish to run. Choosing a file from the browser and closing the dialog box (by clicking on the OK button) loads the file, along with the drive and path, into the file edit window of the dialog box. The user can preset the launch state by clicking in one of the three radio buttons: Normal, Minimized or Maximized. Clicking the Launch button will immediately launch the program. Clicking on Cancel will close the Run box, with no further action.

When files are launched using the Run dialog box, they are recorded in the file dwm.ini. The last file launched using the Run dialog box appears in the file window when the Run box is opened; clicking on the file name will highlight it and it can then be edited. Editing the file in the file window will not destroy it, but push it down on the execution stack. Dwm stores the last twenty applications or files launched using the Run box. Clicking on the down arrow at the right of the file window drops down a list of the twenty applications. Choosing a program or file from the execution stack puts it back at the top of the list; files that are run once from this dialog box eventually drop off the list.

Desktops: This item creates a menu function that calls the dwm desktop editing dialog box. It is through this dialog box that desktops are saved or changed, and pop up menus assigned to left and right mouse clicks. This item provides an alternative way of calling the desktops dialog box; the alternative is to open the dwm options dialog box, and click on the Edit button.

Task manager: This item creates a menu function that calls the Windows Task Manager. The Task Manager is a utility that is used to switch between Widows programs that are currently running, to terminate programs, and to arrange program windows and icons. This utility is useful when users have a large number of programs running on the same screen.

Users who find themselves in the situation in which they have so many programs running in the same space that they need to call the Task Manager regularly are encouraged to use Vern, the dwm virtual environment, as a task switcher. Vern allows users to spread their program windows throughout multiple virtual screens, avoiding the problems of cluttering when many windows are overlapped in the same space.

Refresh left menu: This item creates a menu function that forces dwm to rescan the file for the menu assigned to a left mouse click, and update the menu. This item is most useful when the left menu is assigned to the special Program Manager menu, **progman.mnu**. In this case, the refresh function regenerates the menu by scanning the progman.ini and *.grp files. If the user installs new Windows software, a new program group will often be created during the installation process. Alternatively, the user many wish to put the new programs into a preexisting group, or simply edit the program groups, using the Program Manager. By refreshing the menu, changes to the program manager groups are immediately incorporated into the pop up menu structure.

Refresh right menu: This item is identical to the Refresh left menu item, except it applies to the menu file assigned to the right mouse click.

Save left menu: This item creates a menu function that saves the structure of the left menu to it's menu file. When editing a menu using the dwm menu editing facility, the user will generally save changes to the menu before leaving the editing dialog box. If menus are altered by dragging files from a dob file list window and dropping them on a pop up menu, the new menu structure may need to be saved, or else the changes will be lost when dwm terminates. Note that if the auto save left checkbox in the dwm options dialog box is set, this menu item is not necessary.

Save right menu: This item creates a menu function that is identical to the Save left menu function, except that it applies to the menu assigned to the right mouse click.

Save desktop: This item creates a menu function that saves the configuration of the current desktop to it's .dsk file. When a desktop is saved, the states and positions of all the desktop objects (blanches, dobs, vern and the dwm button) are recorded, along with the environmental variables the user has set for that desktop.

Load desktop: This item creates a menu function that clears the current desktop of all desktop objects, and then reloads it, according to the information in the desktop file. The user may wish to reload the desktop if it is changed during a Windows session, or has been otherwise corrupted.

Clear desktop: This item clears the desktop of all blanches and dobs. The dwm button and vern, the dwm virtual environment, will remain on the desktop, if they were present before it was cleared. The clear desktop function does not affect any open program windows or icons.

Restart Windows: This item creates a menu function that terminates Windows, then automatically restarts it, without returning to the DOS environment. This action is sometimes useful when system resources run low (usually due to badly written programs), or when the system becomes corrupt. This function normally requires a confirmation, as a safety precaution, but the confirmation control can be suppressed by setting the Fast Windows exit checkbox in the dwm options dialog box. Note that certain programs will not allow themselves to be terminated without explicit confirmation, in which case the user may have to shut down programs individually before the restart happens.

Exit Windows: This item creates a menu function that terminates Windows, and returns the user to the DOS environment. This action is normally subject to confirmation; the confirmation control can be suppressed by setting the Fast Windows exit checkbox in the dwm options dialog box.

Exit dwm: This item creates a menu function that terminates dwm. When dwm terminates, all the other dwm applications (dobs, blanch and vern) also terminate. If dwm is being used as the Windows shell, the function behaves identically to the Exit Windows function.

Options dialog: This item creates a menu function that calls the dwm options dialog box. It is through this dialog box that environmental parameters are set, desktops loaded, saved and cleared, menu colours and font chosen, and (indirectly) desktops selected and menus assigned to mouse clicks. The options dialog box is also accessed by double clicking (or right clicking) on the dwm button. Users suppressing the dwm button from the desktop (this is an option set in the dwm options dialog box) should make sure that this function is available in one of the pop up menus.

Editing pop up menus with a text editor

The files **.mnu* files describe user configurable menus. They should be located in the same directory as the rest of the dobs/dwm distribution. These files are ASCII text, and can be easily edited with any word processor or text editor (do not edit the file *progman.mnu*, as any changes you make will be lost when dwm restarts). If you use a word processor, be sure to save the file as ASCII text, otherwise dwm will probably choke on the file.

The .mnu files contain a series of simple lines, each of which begin with a dwm menu *keyword*. The syntax of the dwm menu files is described below and should enable you to customized your pop up menu. Click on keywords to find out more about their function and syntax. See also **Editing menus with the dwm menu editor** for more details, especially for executable items.

Sample menu file:

[dwm]
title=*label for pop up user menu*
separator=
; this is a comment
exec=label, [d:\path\]file.exe, [options, [default_directory, [iconfile,
[iconindex, [minimized]]]]]
menu=*menu label*, flyout
separator=
taskman=*label for task manager item*
desktops=*label for desktops dialog box item*
custom=*label for refresh left menu item*
progman=*label for refresh right menu item*
saveleft=*label for save left menu item*
saveright=*label for save right menu item*
loaddesk=*label for reload desktop item*
savedesk=*label for save desktop item*
options=*label for options dialog box item*
cleardesk=*label for clear desktop item*
restart=*label for restart item*
exitdwm=*label for exit dwm item*
exit=*label for exit item*

[flyout]
title=*label for flyout menu*
separator=
menu=Another flyout,flyout2
menu=Yet another flyout,flyout3
exec=*label*, file.exe, options, default_directory, ,

[flyout2]
...

[flyout3]
...

[dwm] The first line of any menu file **must** be the name of the menu file (without the extension), enclosed by square brackets (of course, you can have commented text appearing above this line in the file). In this case, **[dwm]** indicates that the following menu structure is for the file **dwm.mnu**. The square brackets indicate the start of a menu or submenu section (note that the section [flyout1] toward the bottom of the file, referenced by the menu= keyword in the example).

; dwm ignores all text appearing after a semicolon, to the end of the line. Use comments to make notes to yourself, or "remove" a line by commenting it out, instead of deleting it.

exec= This keyword will create a menu item for an executable program; clicking on the menu item will launch the program. Menu items generated by the exec= keyword can also be blanched. Each entry generated with exec= must have a label and executable file specified. The other entries (options, default dir, iconfile, iconindex, minimized) are **optional**. The structure for this keyword is shown below, click on various sections of the string for more information about the function they serve (you can also refer to the example shown below. Note that the brackets [ ] should not be used; we include them here to indicate optional parts of the string.

exec=*label*, [d:\path\]file.exe, [options, [default_directory, [iconfile, [iconindex, [minimized]]]]]

example: exec=Edit menu file, c:\wordapps\texted.exe, dwm.mnu, c:\dwm,c:\icons\ iconpak.dll,475,1

- this example will create a menu item with the label "Edit menu file", clicking on this item will launch the program texted.exe (a hypothetical text editor), located in the hypothetical wordapps directory on the hypothetical c: drive. The file dwm.mnu is specified as an option, it is loaded into texted.exe automatically when the application executes. The default directory is given as c:\dwm. The icon appearing on this menu item will be the 475th icon in the hypothetical iconfile *c:\icons\iconpak.dll*. The minimized option (the last option) is specified; when the program starts it appears as an icon on the screen.

label: text the user wishes to appear on the menu for the executable. The text has nothing to do with the executable file specified immediately after the *label* in the **exec=** string, so the user can create somewhat whimsical menu items.

[d:\path\]file.exe: the name of the executable, with the extension. It may also be necessary to include the drive and path specifiers for the program (i.e.: c:\stuff\secret\myprog.exe). Note that the menu item will appear more rapidly if you include the FULL path to the executable containing the icon, and that if you omit the extension the icon will NOT be found.

options: this entry may contain the name of a file (or files) to be loaded into the executable at launch, or may contain command line switches.

default_directory: specifies the default directory to be used by the executable program.

iconfile: gives a path to the file containing the desired icon (icons are most commonly stored in files ending in *.ico*, *.dll* or *.exe*).

iconindex: the index, starting from zero, into the iconfile for the desired icon.

minimized: To start the program in a minimized state set this option to 1, a value of 0 will start the program in a normal manner.

menu= This keyword creates a menu item which spawns a flyout menu when selected. The first entry for this keyword is the *label* for the menu item, the second entry refers to a *section identifier* in the dwm.mnu file where information for the flyout menu is found (this referenced section must be enclosed by square brackets). Note that menus may be nested, that is, you may have flyout menus containing other flyout menus containing other flyout menus ...

[flyout] This indicates the start of a flyout menu section, referred to by the second parameter of a **menu=** statement somewhere in the dwm.mnu file, and must be enclosed with square brackets (make sure there are no spaces between the section identifier and the brackets). All menu keywords between this section identifier and the next will be included in this menu section and may contain any of the menu keywords (including the menu= keyword, for nested flyout menus).

Modifying a pop up menu with dobs

Users can take advantage of the drag and drop aspects of **dobs** (the dwm file manager) to quickly load executable files into dwm pop up menus. This is undoubtedly the best way of incorporating executable items into dwm menu structures.

Start by opening a dwm pop up menu (**left** or **right** click on the desktop), and open any flyout menus, if necessary. The user may wish to move the target menu or submenu to a convenient screen location. Menus are moved by left clicking on the menu title bar and **dragging** the menu to a desired position.

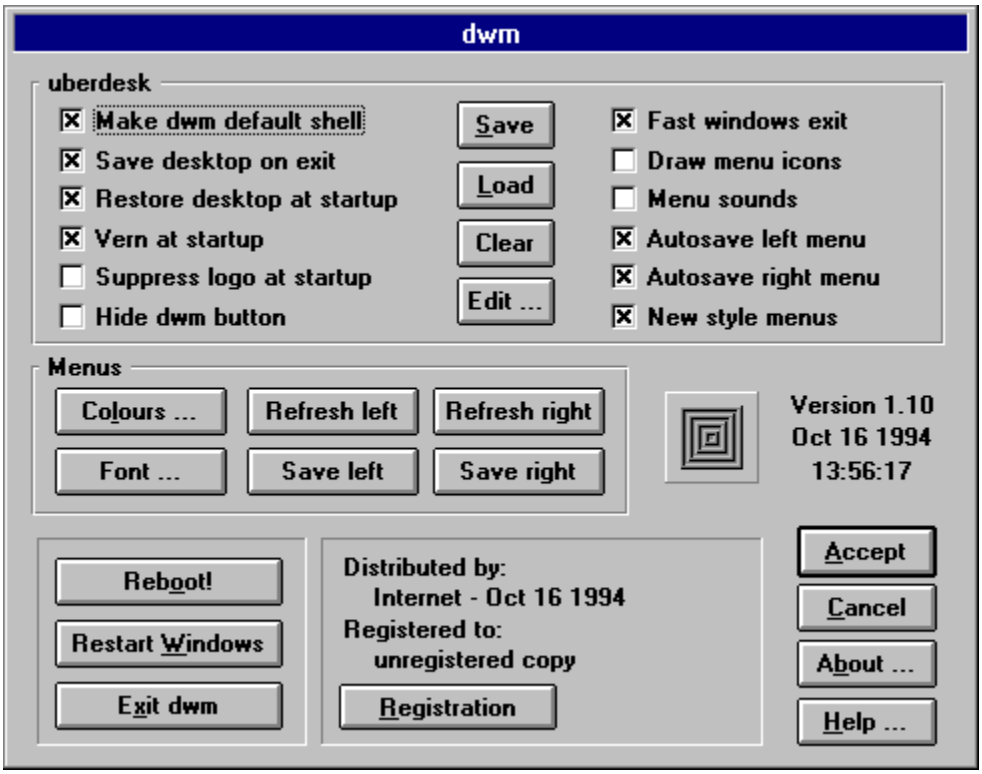
Next, open a dob file list window, and select an executable file by left clicking on the file name (executable files are coloured green, by default, in a dob file list window, if the show colour option in the dobs options dialog box is set). Once the file is selected, left click on the file a second time and hold down the mouse button, and **drag** the file out of the dob window. The mouse cursor will change to a small file icon as the cursor leaves the dob window. The file can now be dropped onto any of the menus or submenus, and the file will automatically be incorporated as an executable menu item (the file gets inserted in the menu cell immediately below the drop point).

Only executable files, or files that have extensions **associated** with executable programs, can be loaded into menus. In general, any file that can be blanched can be loaded into a dwm pop up menu. Files must be loaded into menus individually (multiple file selections cannot be loaded into a menu simultaneously).

Note that when menus are loaded with executable items by drag and drop from dobs, the information fields for the executable item contain certain defaults. In particular, item label is the name of the dropped file, the default directory will be the same as the directory of the dropped executable program, the options line is left blank, and the icon for the item will be it's normal icon. The user may change the label, default directory, and icon, and supply options or command line switches by editing the item with the **dwm menu editing facility**.

The dwm button and the dwm dialog box

Double clicking with the left mouse button (or **right clicking**) on the dwm button (shown above) opens the dwm options dialog box. The dwm dialog box is shown below. Click on various parts of the box for more information about their functions.



Desktop description: The current desktop descriptor is shown near the top of the dialog box. The description is a sort of mnemonic for all the elements that make a particular desktop unique, including the desktop objects (along with their positions and states), the pop up menus assigned to left and right mouse clicks, and the environmental variables that control the way the menus and dwm behave.

Make dwm default shell: Checking this item and clicking on Accept modifies the Windows *system.ini* file, specifying dwm as the shell. The old shell is preserved by the line oldshell= in the system.ini file, and is restored as the shell when this option is deselected. Whenever the user toggles this option, they must either reboot the computer or restart Windows for the change of shell to take effect.

Save desktop on exit: If this option is checked, dwm updates the current desktop file when dwm is exited, or the Windows session ends. Desktop saves preserve the states and positions of all desktop objects (vern, blanches, dobs and the dwm button). If this option is off, the current desktop file is not updated to include the changes to the desktop environment that may have occurred during a given session.

Restore desktop at startup: If this item is checked, the current desktop is restored when dwm starts. The presence or absence of vern and the dwm button are controlled by the state of the hide dwm button and vern at startup switches. The current desktop is specified by the desktop= line in the dwm.ini file, and is usually the desktop that was in use when dwm was last exited, or the previous Windows session was terminated.

If the Restore desktop at startup option is not checked, the desktop will be clean when dwm starts. This means that no blanches or dobs will appear, and the presence of the dwm button and vern will depend on whether the user has opted for their inclusion at startup (hide dwm button and vern at startup are other options set in the dwm dialog box).

Vern at startup: If checked, the matrix of virtual screens making up vernspace is run automatically when dwm is run (or when Windows starts, if dwm is the shell).

Suppress logo at startup: This item is pretty much self explanatory.

Hide dwm button: The dwm button is intended to be an integral part of the desktop, but certain users may wish to suppress its appearance on the desktop. The button can be suppressed from the desktop at startup by setting the hide dwm button option in the dwm options dialog box. If the dwm button is absent from the desktop, access to the dwm options dialog box can be gained through a pop up menu, if it contains the Options dialog menu item.
The dwm button can be removed at any time by hitting **Ctrl+Shift+h** when dwm has focus, and can be restored by repeating the **Ctrl+Shift+h** sequence, when dwm has focus. Dwm has focus (i.e.: is the active program) when using pop up menus or the dwm options dialog box (and immediately after closing a pop up menu, or closing the dialog box), or when the dwm button is moved or clicked on.

Fast Windows exit: The actions of exiting Windows, terminating dwm or restarting Windows are normally subject to confirmation, that is, a dialog box pops up asking the user if they are sure they want to exit, restart etc. Although such confirmations are intended as a safety measure, certain users may wish to suppress the confirmation control by setting this option.

Draw menu icons: When dwm menus pop up, executable menu items are normally painted with the program icon, an icon chosen by the user for that particular item, or a default icon supplied by dwm. Painting icons on the pop up menu items can take some time, since the icon must be extracted from the specified icon file. The extraction takes even longer if the full path to the iconfile is not supplied (even if the iconfile is in a directory included in the PATH statement in the autoexec.bat file). Users wishing to suppress the icons, for aesthetic reasons or for quicker menu draws should turn this option off.

Menu sounds: This is an option that plays sounds when popping up a dwm menu. The sound played is the .wav file assigned to the asterisk event, set through the Sound utility of the Windows Control Panel.

Autosave left menu: This option will force dwm to update the menu file assigned to the left mouse click whenever the menu is modified with the menu editor, or by dragging and dropping executable files from a dob window.

Autosave right menu: This option works the same way as the Autosave left menu option, except it applies to the menu assigned to the right mouse click.

New style menus: This toggle is used to switch between the classic style dwm menus and the new menus. Classic menus have highlight boxes drawn around each cell in the pop up menu, while the new style menus have items arranged on a smooth background.

Save desktop: Clicking on this button saves the desktop, preserving the particular combination of desktop object (vern, blanches etc.) states and positions, along with the environmental variables, to the current desktop file. The user may wish to save a desktop if they plan to switch to another desktop during a session, so that any changes made to the objects on the desktop are recorded. The desktop also can be saved by hitting **Ctrl+Shift+s**, when dwm has focus (is the active application).

Load desktop: Clicking on this button restores the currently selected desktop, according to the information in the dsk file for that desktop. If changes are made to the desktop objects, and the changes are not saved, clicking on this button will clear the desktop and restore it to its original state. The user can alternatively load the desktop by hitting **Ctrl+Shift+l**, when dwm has focus (is the active application).

Clear desktop: Clicking on this button clears the desktop of all blanches or dobs currently on the desktop, without saving to the desktop file (the blanch and/or dobs options are preserved, if new ones are generated). Vern, and the dwm button will remain on the desktop when it is cleared, as will any open program windows or icons. If you unintentionally clear the desktop, it can be resurrected if you click on the Load button. The desktop also can be cleared by hitting **Ctrl+Shift+c**, when dwm has focus (is the active application).

Edit...: Clicking on the Edit... button opens the **dwm desktops editing dialog box**. It is through this dialog box that users can create new desktops, assign descriptions to desktops, switch between available desktops, and change the menus assigned to the left and right mouse button clicks. The desktops dialog box can also be accessed through a dwm pop up menu, if it includes the Desktops menu item.

Colours Clicking on this button opens a dialog box that allows the user to change the default menu **colours** (colour of the text and background and cell highlights of the pop up menus)

Font: Clicking on the Font button opens up a dialog box in which the user can select the font, size and style for the text appearing in the dwm pop up menus. Note that the although a text colour option is available in the font dialog box, any selection will be ignored. Users wishing to change the text colour should use the Colours... option in the dwm options dialog box.

Refresh (left): Clicking on a Refresh button forces dwm to rescan the file for the menu assigned to the mouse click, and update the menu. This item is most useful when the left menu is assigned to the special Program manager menu, progman.mnu. In this case, the refresh function regenerates the menu by scanning the progman.ini and *.grp files. If the user installs new Windows software, a new program group will often be created during the installation process. Alternatively, the user many wish to put the new programs into a preexisting group, or simply edit the program groups, using the Program Manager. By refreshing the menu, changes to the program manager groups are immediately incorporated into the pop up menu structure. The left menu file can also be refreshed by hitting **Ctrl+Shift+n**, when dwm has focus.

Save left: Clicking on this button saves the structure of the left menu to it's menu file. When editing a menu using the dwm menu editing facility, the user will generally save changes to the menu before leaving the editing dialog box. If menus are altered by dragging files from a dob file list window and dropping them on a pop up menu, the new menu structure may need to be saved, or else the changes will be lost when dwm terminates. Note that if the auto save left checkbox in the dwm options dialog box is set, this button need not be used.

Refresh right: This button performs the same function as the Refresh left button, except that it applies to the menu file assigned to the right mouse click. The right menu file can be refreshed by hitting **Ctrl+Shift+m** when dwm has focus.

Save right: This button performs the same function as the Save left button, except it applies to the menu assigned to the right mouse click.

Accept: Validates an changes made to checkbox items. If no changes have been made, clicking on accept simply closes the options dialog box.

Cancel: Closes the dwm options dialog box. Any changes made to checkbox items will not be preserved if cancel is selected.

Help...: Launches the help system for the various applications in the dobs/dwm package.

About...: Displays information and quick tips for the various applications in the dobs/dwm package.

Reboot!: Reboots the computer (warm boot, similar to holding down **Ctrl**+**Alt**+**Del** together). If the **fast Windows exit** option in the dwm options dialog box is set, dwm will not prompt the user to confirm the reboot.

Restart Windows: Restarts Windows without rebooting. If the **fast Windows exit** option in the dwm options dialog box is set, dwm will not prompt the user to confirm the restart.

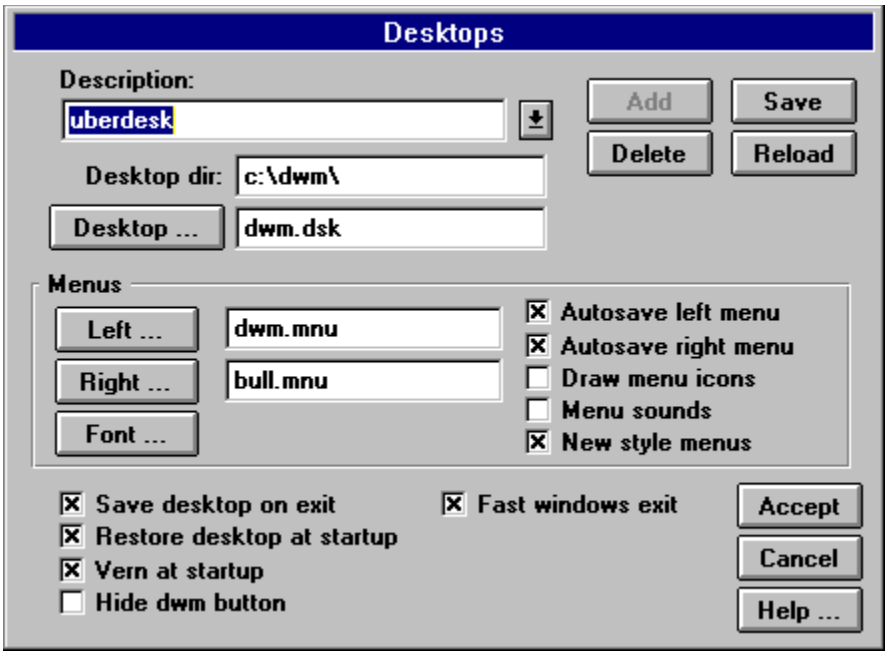
Exit dwm: Closes dwm. If dwm is the shell, Windows is also terminated. If the **fast Windows exit** option in the dwm options dialog box is set, dwm will not prompt the user to confirm exiting Windows.

Registration stamp area Displays the name of the licensee and their registration code. Shareware copies of dwm are unregistered. Clicking on the registration button opens a dialog box, where the user can enter a registration code, provided by GPG. After proper registration, the register button disappears. Click on the **Register** button at the top of the help screen for more information on registration.

Version and compile date: Displays the dwm version number, as well as the compile date. If the date is more than a year old on the shareware copy you wish to register, contact us first before you order (click on the **register** button at the top of the help screen for more information.

The desktops dialog box

The desktops dialog box is accessed by clicking on the **Edit...** button in the **dwm options dialog box**, or through a dwm pop up menu that includes the **Desktops** menu item. It is through the desktops dialog box that the user can change desktops, create new desktops and assign menu files to left and right mouse clicks. The desktops dialog box is shown here. Click on various parts of the box to find out about the functions they serve.



Description: The desktop description is used as a descriptive text string; a mnemonic for all the elements that make up the desktop. A desktop is defined by the desktop objects (along with their positions and states), the pop up menus assigned to left and right mouse clicks, and the environmental variables that control the way the menus and dwm behave. In the dwm options dialog box, it is the desktop description, and not the desktop file name, that appears at the top of the box housing the desktop options.

Description edit window: The description edit window is also a drop down list box; clicking on the down arrow drops down the list, revealing all the available desktop descriptions.

The description entered into the edit box becomes associated with the desktop file appearing in the Desktops edit window, along with the particular combination of assigned pop up menus and the environment variables set in the checkboxes. The user can thus have many desktops (referenced by the descriptions) that each use the same desktop file (i.e.: will have the same arrangement of blanches and dobs), but have different environmental parameters, or have different menus assigned to left and right mouse clicks.

The description list can be used to change desktops; click on the down arrow to reveal the list of descriptions and click on one of the descriptions on the list. Choosing a new description from the list will change the other information in the dialog box.

To create a new desktop, start by adding a new description to the description list. To do this, click in the Description window (whatever description is in the window will become highlighted) and type in a new description (note that the description being typed over is not destroyed, just pushed down on the list). Click on the Add button, and the description is added to the list. The user can now assign a desktop file, left and right menu files, and set the environmental variables, in order to completely define the new desktop. Click on the Save button to save the description and record the particulars for the desktop.

Desktop directory edit window: This edit window is used to specify the full path to the directory containing the desktop file appearing in the Desktop edit window. By default, it is c:\dwm\, the default directory created for the entire dwm distribution during installation of the software package.

Desktop file edit window: This edit window is used to specify the desktop file for the desktop. Desktop files, which have the extension .dsk, contain instructions for loading blanches and dobs (vern and the dwm button are also desktop objects, but are handled differently). Different desktop files will have different collections of these desktop objects, and will also record the various user specified blanch and dobs options.

When dwm is first installed, there is only one desktop file available, dwm.dsk. The user is free to create as many desktop files as are necessary, each with their own unique collection of blanches and dobs. Switching desktop files during a dwm session will clear the desktop of all dobs and blanches, and reload these objects according to the new desktop file.

Add button: Click on the Add button to add a new desktop description to the list of desktops. This button becomes active only when a new desktop description is entered into the Descriptions edit box (to enter a new description, click in the edit box, and type in a new description string). When a desktop description is added, dwm creates a new section in the file desktops.ini (located in the WINDOWS directory), headed by the newly added description. Each section of the desktops.ini file defines a complete desktop, listing the desktop file directory, desktop file, left and right menu files and environmental option switches that make the desktop unique.

Delete button: Click on the Delete button to remove the description appearing in the Description edit window from the description list. When a description is deleted, the section headed by that description in deleted from the desktops.ini file. Note that dwm will not allow the user to delete all the desktop descriptions, there must always be at least one available.

Save button: Click on the Save button to save the desktop description, along with the information linked to that description (i.e.: all the other stuff shown in the desktops dialog box: desktop directory, desktop file, left and right menu files and environmental options). When the desktop description is saved, the desktops.ini file is updated to include any changes that have been made to the desktop variables.

Reload button: Click on the Reload button to reload all the desktop information recorded for the current description into the desktops dialog box. This button effectively undoes any changes made while editing the desktop variables (desktop file, left and/or right menu files, environmental variables etc.), and resets them all to their original state (i.e.: the way they were before changes were made).

Menu section of the desktops dialog box: The menu section of the Desktops dialog box provides controls for assigning different pop up menu files to left and right mouse clicks, and defining environmental variables specific to menu actions. When dwm is first installed, there are only two menu files available, dwm.mnu and progman.mnu. The file dwm.mnu is assigned to the left mouse click, and contains some stock Windows applications, with a few native dwm functions. The right mouse click is assigned to the file progman.mnu, which is a list of the Program Manager groups.

Menus assigned to left and right mouse clicks must be valid dwm menu files (these have the extension .mnu). New menus can be created, altered or otherwise modified by using the dwm menu editing facility.

Left menu file edit window: To change the left or right menu assignments, enter a new menu file name into the appropriate menu file edit window (there is a menu file edit window for each of the left and right menu file assignments). Editing is done in the normal way: left click on the file name (the cursor becomes an I shaped editing cursor), and edit the file name, or double click on the menu file (it will become highlighted), and type in the complete name of a menu file including the extension. Alternatively, click on either of the Left or Right buttons to open up a file browsing dialog box. Use the file browser to select the desired menu file, and click on the OK button to close the browser (the selected menu file will be loaded into the appropriate edit window when the browser closes).

Right menu file edit window: To change the left or right menu assignments, enter a new menu file name into the appropriate menu file edit window (there is a menu file edit window for each of the left and right menu file assignments). Editing is done in the normal way: left click on the file name (the cursor becomes an I shaped editing cursor), and edit the file name, or double click on the menu file (it will become highlighted), and type in the complete name of a menu file including the extension. Alternatively, click on either of the Left or Right buttons to open up a file browsing dialog box. Use the file browser to select the desired menu file, and click on the OK button to close the browser (the selected menu file will be loaded into the appropriate edit window when the browser closes).

Font... button: The Font... button in the Menus section works in the same way as the Font... button in the dwm options dialog box. Clicking on the Font... button opens a common Windows dialog box, in which the user can set the font, size and appearance for the text on the pop up menus.

Environmental variables: When the desktops dialog box is opened, it shows the states of the environmental variables for the current desktop. Remember, the current desktop is the sum of the desktop file, menu assignments and environmental options, and is defined by the desktop description. The environmental variables are grouped into two areas; menu specific options are grouped in the Menus section, and the dwm desktop variables are grouped at the bottom of the dialog box.

When the Desktops dialog box is initially opened, these variables are obviously those currently in use, since the dialog box always opens showing the information for the current desktop. If the user switches desktops (i.e.: by selecting a new description from the drop down descriptions list), the environmental variable checkboxes will change to show the settings defined for the alternate desktop. The user can set environmental variables for any or all of the defined desktops from within the Desktops dialog box.

Note that two of the environmental options, Make dwm default shell and Suppress logo at startup, are not available in the Desktops dialog box. This is because these two options are global, that is, common to all available desktops. Changing either of these options for one desktop forces a similar change for all the other desktops.

Accept button: Click on the Accept button to validate all changes, and close the Desktops dialog box. The changes (desktop, desktop file, menu assignments, environmental options) are incorporated immediately, without having to restart dwm or exit Windows.

Cancel button: Clicking on the Cancel button closes the Desktops dialog box, ignoring any changes that may have been made.

Help button: Click on the Help button to launch the dwm on-line help system starting up in the section on the Desktops dialog box (i.e.: no need to negotiate the help menus to find the section on this dialog box).

Setting menu colours

Clicking on the **Colours...** button in the dwm dialog box opens another dialog box, in which the user can specify colours for the dwm pop up menus. The dialog box is shown below; click on various parts of it to find out more about what function they serve...

Note: the colours are specified by hexadecimal triplets representing RGB values. Each six digit number is composed of three two digit numbers, ranging from 00 to FF. The 16 'pure' colours are listed below, with their corresponding hexadecimal codes.

000000 = Black
FFFFFF = White
800000 = Dark Red
FF0000 = Red
008000 = Dark Green
00FF00 = Green
000080 = Dark Blue
0000FF = Blue
800080 = Dark Magenta
FF00FF = Magenta
808000 = Dark Yellow
FFFF00 = Yellow
008080 = Dark Cyan
00FFFF = Cyan
808080 = Dark Grey
C0C0C0 = Light Grey

Light shade: This colour is for the light highlight ringing a menu button, used in combination with **dark shade** to give the buttons a 3-D look.

Colour: Click on a button to select colours using the choose colour common dialog box. Note that it is usually best to select solid colours, especially if you have a 16 or 256 colour display driver.

Background: Colour of the body of the dwm pop up menu. You should make sure it is somewhat different from the text colour, or you won't be able to read any of the menu entries.

Dark shade: This colour is for the dark highlight ringing a menu button and used in combination with **light shade** to give the buttons a 3-D look.

Text colour: Colour of the text in a dwm pop up menu. You should make sure it is somewhat different from the text colour, or you won't be able to read any of the menu entries.

Light grey button: Sets the colours to a system default, with black text on a light grey background.

Dark grey button: Sets the colours to a system default, with black text on a dark grey background.

Accept button: Validates any changes to the colour scheme, and closes the dialog box.

Cancel button: Closes the dialog box, ignoring any changes to the colour scheme.

Creating a blanch from a menu item

A dwm menu item can be **dragged** from a pop up window and dropped onto the desktop, where it leaves a blanch (button launch). In order to become a blanch, the item must be an executable program, or a file with an extension **associated** to an executable. Blanches are useful as deferred execution sites, and provide a quick way of changing the command line or options for the program dragged off the pop up menu (see also: **creating blanches**).

Tips on using dwm

- **Double click** (or **right click**) on the dwm button to open the **dwm options** dialog box (**Alt+Enter** will also open this dialog box, if the dwm button is active).
- Make dwm the **Windows shell** after installation.
- Other dwm applications (vern, dobs, blanch) will not work without dwm.
- **Left click** on the desktop (or hit **Ctrl+l**) to pop up user configurable menu.
- **Right click** on the desktop (or hit **Ctrl+r**) to pop up a different menu (initially the program manager group menu, by default).
- **Left click** on a menu item to execute the program or function.
- **Right click** on a menu item to open the dwm menu editing dialog box..
- Drag an executable item off of a pop up menu and drop it on the desktop to create a blanch for that item.
- To move the dwm button, simply click left and drag the button.
- Dwm keeps track of all desktop objects (vern, blanches, dobs and the dwm button) between Windows sessions
- Change the **startup=StartUp** entry in the dwm.ini file to change the default name of the Progman startup group (useful for non-English Windows versions).
- Use the following keyboard shortcuts:

Ctrl+Left click	Open Windows Task Manager
Ctrl+Right click	Open dwm Options dialog box
Ctrl+Shift+c	Clear desktop
Ctrl+Shift+s	Save desktop
Ctrl+Shift+l	Load desktop
Ctrl+Shift+n	Read the left click menu file
Ctrl+Shift+m	Read the right click menu file
Ctrl+Shift+h	Show/hide dwm button

Installing dwm in a Network environment

To use a copy of dwm that has been installed to drive **z:\dwm** on a network server:

- copy the files **dwm.mnu** and **dwm.ini** to your personal Windows directory in which you have write access (for example, **g:\windows**). If you do not have the file dwm.ini you can create one with any ASCII text editor and specifying [dwm] on the first line of the file.
- edit the dwm.ini file, with an ASCII text editor, and change (or add if you have just created the file) the entry **dwmdir=z:\dwm** to the directory where you have copied the files dwm.mnu and dwm.ini, for example **dwmdir=g:\windows**
- run Windows, select Run from the Program Manager File menu and run *z:\dwm\dwm.exe,* or wherever it happens to be installed.
- If you use dwm as the default Windows shell make sure that the network directory in which the dwm distribution is installed is in the *PATH* environment variable (e.g. PATH=z:\windows;z:\dwm;...).
- System Administrators should make sure that the dwm directory has the shared attribute set for all the files in the directory in which the dwm distribution is stored.
- If you have problems running dwm on a network please feel free to contact GPG (see **Product Support** in the register.hlp file).

Left click: Position the mouse pointer over the object to be *clicked* then press and release on the left most mouse button. **Southpaw note:** if have swapped the left and right mouse buttons in the *mouse applet* of the **Windows Control Panel** you may consider a *left click* to be the right most mouse button.

Right click: Position the mouse pointer over the object to be *clicked* then press and release on the right most mouse button. **Southpaw note:** if have swapped the left and right mouse buttons in the *mouse applet* of the **Windows Control Panel** you may consider a *right click* to be the left most mouse button.

Double click: Position the mouse pointer over the object to be *double clicked* then press and release on the left most mouse button twice in quick succession. **Southpaw note:** if have swapped the left and right mouse buttons in the *mouse applet* of the **Windows Control Panel** you should use the right most mouse button when performing a *double click*.

Dragging: Position the mouse pointer over the object to be *dragged* then press and **hold** the left most mouse button down and move the mouse across the desktop. If the object is dragable it will move across the screen with the cursor; often the mouse cursor changes shape to indicate that something is being dragged. To drop whatever it is that you're dragging, just release the left mouse button. **Southpaw note:** if have swapped the left and right mouse buttons in the *mouse applet* of the **Windows Control Panel** you should use the right most mouse button when performing a *mouse drag*.
Note: Many of the dwm desktop programs (vern, blanch and dobs in the button state) require that you hold down the **Ctrl** key before dragging the program around the desktop. This requirement simply prevents the objects from *sliding* around on the desktop.