



Chapter One

What's New for Opus 5.5

We thank you for your continuing support of Directory Opus. After more than a year of extra development since its original release, Opus 5.5 contains many changes and enhancements over the original Opus 5. Thanks to the feedback from many loyal users and a dedicated band of Beta testers, we are confident that you will enjoy this new enhanced version of the Amiga world's most popular directory and file management utility.

With version 5.5, almost every facet of Opus has been examined and subjected to user scrutiny, backed up by extensive field testing by very persistent Beta testers. In fact, Opus 5.5 is more of a new program than an update. We believe that this version provides many significant improvements in both operational power and usability.

Opus 5 was the first program to attempt to integrate the concept of a file manager with that of a replacement Workbench. We have worked through a number of compatibility problems this caused and this idea has been greatly extended in Opus 5.5. We now believe that this is the best method of using Opus. Why run workbench AND a file manager when Opus integrates the two!

A brief summary of the most important changes is provided below and the details have been incorporated into this new manual.

Highlights of Directory Opus 5.5

- A new **Icon Action Mode** gives all the power of Name Mode but with icons. You can now use buttons and commands with icons.
- Button banks and Listers need no longer be activated first in order to see right and middle mouse clicks. You can select a button without first activating the underlying window.
- You may now have **borderless Button banks** with a sleek minimal dragbar instead of a full window border.
- New **Filetype-specific pop-up menus** allows special menus for icons and files. Through Opus Filetypes you can now add custom menus for different types of files and icons.
- Custom buttons now have a pop-up menu giving access to an extended selection of commands for the left mouse button events while retaining fixed commands for middle and right mouse clicks.
- Independent **HotKeys** are now supported and these may be local to Opus or operate globally on your system.
- A new **Scripts** system allows functions to be executed upon certain events including a double mouse click, disk inserted or removed, etc.
- Custom menus have been improved and multiple user menus are now supported. All menus may now have sub items.
- **WorkBench Replacement Mode** has been enhanced. Opus 5.5 can now be used with complete confidence to replace the standard Workbench.
- An integrated **OpusFTP** capability lets you seamlessly access remote Internet sites from standard Opus Listers just as if they were local directories.

- A new ***Automatic Filetype Creator*** allows you to create and test Filetypes with ease.
- A new ***font viewer*** is now included. Just double-click on a font to view it.
- Enhanced Name Mode Listers now have ***field titles, single-click re-sorting by fields***, plus a new version field which reads the internal version information from each file.
- Listers can now have individual toolbars.
- New ***colour remapping*** of button and icon images provides support for 'Magic Workbench' and other palette changing systems.
- A new ***internal Opus CLI*** gives immediate access to internal commands allowing to you quickly test commands and ARexx scripts.
- Unwanted drive icons can be selectively hidden from the Opus main window.
- Dynamic drag-selection of icons provides better performance.
- Enhanced ***clipboard*** support provides full cut, copy and paste with all string gadgets, and, within Listers, the names of selected entries can be copied to the clipboard.
- Listers are no longer blocked while busy - you can now resize, iconify, and scroll busy Listers.
- The '*Execute Command*' requester now remembers its last 20 commands entered.
- Icon and Lister snapshots are now stored separately from Workbench.

What's New

- Listers in Icon or Icon Action Mode can now display a ***background picture or pattern***.
- The Lister status bar details can be customised.
- Opus 5's text output can be directed to your selected device other than 'CON:'
- Icon borders and names can be disabled for individual icons.
- Buttons in banks and toolbars can now use two-frame images.
- Several new internal commands have been added.
- Many new ARexx commands have been added and old ones have been extended with new features. You can now even add you own internal commands directly to the Opus command list.



Chapter Two

Introduction and Overview

Thank you for purchasing Directory Opus 5. We believe you will be impressed by its new power and features.

This manual has been designed to lead you through using Opus 5, or allow you to quickly skip to chapters of interest. This chapter is designed to tell you something about the concepts behind Directory Opus 5. It will provide you with a general overview of its operation, so you can start using the program immediately. Even if you read nothing else in this manual, you should read this chapter! Chapter two provides you with a simple introduction to the Amiga filing system while the subsequent chapters discuss the individual parts of the Directory Opus 5 system in more detail.

Directory Opus 5 - The Power of the Amiga Realised!

There are now many directory utilities for the Amiga, but nothing like Directory Opus 5. Whatever program you used before, Opus 5 heralds a totally new generation of directory utilities. Directory Opus 4 reached the effective limit of power and flexibility for a static directory utility program. Opus 5 breaks out of the mould! It uses the power of the Amiga in a way rarely seen before, giving you the most powerful Amiga disk utility ever.

There are directory utilities, then there's Directory Opus 5!

Directory Opus 5 is the result of a total rethink on the nature of directory utilities from authors who have been developing Amiga software for many years. By using a strict object orientated design

methodology to harness the often hidden, multi-tasking power of the Amiga Operating System, we have been able to create a totally new program, which is much smarter than Opus 4 in the way it delivers this new power and flexibility.

Workbench Replacement

When used as a **Workbench replacement**, Opus 5 greatly enhances your productivity since there is no need to keep swapping between Workbench and your file manager. Opus provides a seamless integration of file manager and Workbench functions.

Although Opus 5 is simple and easy to use, it still gives you an extensive ability to configure the display and program operations exactly as you desire. And, all this is done in full compliance with the Amiga Style Guide principles.

Multitasking as it should be done!

Opus 5 is a fully multi-threaded, internally multitasking suite of programs, which are called into operation when required. These programs invisibly control the operation of each system object, whether it be the visual display of a directory list, or a specific action such as copying files. Each visible component of the Opus 5 display, whether the main backdrop window, a directory Lister, a Button Bank, or other object, is actually controlled by an independent program task which interacts with the other objects in the system as required. Commands can pass instructions to source and target objects to perform actions on demand, independently of the other objects of the visual display.

So what does all this technical design talk mean to the user? Simple! Opus 5 now has more power than ever before, but is even easier to use!

No Waiting, No Delay

The object design concept, with its inherent multi-tasking, is what actually gives Opus 5 its impressive power and makes it so fast

and efficient. Once you understand how Opus 5 works, you will no longer need to wait while one action finishes before starting another. For example, while de-archiving into one directory, there is now no need to wait for this to finish before doing something else. Once the action has been launched, you can immediately open a new directory Lister and start performing other tasks, all while the first task completes. Similarly, you can download a file from a remote Internet site using the internal OpusFTP without blocking other activities.

The power of Opus 5 is also demonstrated when editing any of the Opus 5 objects or system configuration items. The configuration items and actions of each object can be edited separately and independently, while never blocking the actions of other objects. So, while you are editing the commands attached to a set of custom buttons in a button bank, you can still be performing jobs with other component objects such as the file Listers.

"Superfluous Glitz"

It has been said in some quarters that the Amiga user does not need all this "*superfluous glitz*" in a file manager. Some people seem to think that the way we all did things back in the heady days of 1988 with WB1.3 should be good enough for everyone. We beg to differ! The Amiga has advanced and so have its developers. We now know how to harness some of the machine's native multitasking ability for our own uses.

Opus 5.5 adds even more configurability and user options to give you the power to run the system just the way you require. No longer are you, the Amiga user, confined to a simple two-windowed display. Opus 5 uses the full potential of the Amiga OS to provide rapid access to an unlimited number of directory displays, button banks and icon/image banks. If the power of this approach was not amply demonstrated in Opus 5, it is now even more evident with Opus 5.5 and its in-built OpusFTP commands which allow you to seamlessly access Internet sites just as if they were local directories.

You may have to adjust your brain to new concepts, but the extra productivity gained will be well worth it.

The Opus 5 Visual Display Objects

Opus 5 can be run with a myriad of different configurations for almost every conceivable use. However, the essential nature of the Opus 5 display consists of a few simple component windows and objects :-

The **Main Window** is the parent window of the Opus 5 system. It displays icons representing selected devices, Opus 5 groups and any left out icons. This may be opened on any public screen in your system including Workbench. It provides access to all other objects in the Opus 5 system. If the main window is opened on the Workbench Screen, Opus can be run in its most powerful mode as a full Workbench replacement.

Listers are independent windows which display lists of files and directories. Have only one Lister open to view contents of a disk or have as many as you desire. Each Lister may be a source or destination for actions and you may have multiple sources and multiple destinations if desired. Listers may show the file list using either text or icons and you may optionally have special toolbar for each Lister.

Button Banks are windows which display custom action buttons showing text or graphic images.

Other independent windows are provided to allow you to customise the visual display and procedural operations of Directory Opus 5 and add or edit **HotKeys, Scripts, Filetypes** and many other configurations. These aspects of the program may be changed at any time while the program is performing other tasks.

Apart from the Main Window, each of the above components is actually the visual footprint of a completely separate program task, which is invoked only when required. There can always be only one Main Window, but, at any given time, you may have none or any number of Listers and Button Banks in any configuration.

The Main Window

When you run Opus 5, the first component opened is the Opus 5 Main Window. This is the foundation object in the Opus 5 system. From here, you can easily access all the volumes and directories in your system, launch the other Opus 5 components, and, from the menus, hot keys, and double left, right, and middle mouse button clicks, you can create, edit and adjust Lister displays, Button Banks, and other items which control the Opus 5 configuration system. Initially, control over Opus 5 is provided by global menus from this Main Window. These actions are fully explained later.

The Opus 5 Main Window is much more than just a simple place holder window. The concept is very similar to that of the standard Amiga Workbench window with which you are familiar. The Opus 5 Main Window displays the disk and device icons for all the volumes in your system. (You can also decide which icons *not* to show.) Just like Workbench, you can '*leave out*' your favourite directories, files, and programs. Unlike Workbench, Opus provides the concept of '*Program Groups*' to help you organise your favourite programs more easily. Opus 5 also supports full *drag and drop* actions for *all* objects on the Opus Main Window.

One of the powerful options provided by the Opus 5 system is the ability to use the Opus 5 Main Window as a *complete Workbench replacement* - so you never need to run the Workbench program itself! The Opus 5 main window provides all the functionality of Workbench, but with the extra power of Opus 5.

File Listers

The working heart of the Opus 5 system is the file Lister window. This is used to display a list of directories and files in the order and format you desire. Traditionally, one uses two Listers, one as a source and one as a destination, when copying files between directories. But, often only one Lister is required, for example, when you wish to view and delete files from a specific directory. On other occasions, you may wish to copy files to more than one destination or compare files in multiple directories. Opus 5 gives you the flexibility to use as few or as many Listers as you require to get the job done.

As one of the axioms of Opus 5's object orientated design, file Listers are designed to be dynamic. Do not consider them as the old-style, static file display windows which you must leave open and on screen all the time! Each Opus 5 Lister is a fully independent program with its own in-built functionality. They have been crafted to be transient objects, to be brought into existence for the specific job in hand, then discarded. Alternatively, if your application requires it, you can readily create a dual or multi-Lister display, lock the Listers in place, and save the complete configuration setup for use at a later time.

Lister Display Modes

A file Lister can show files in one of three modes, **Icon Mode**, which displays icons just like Workbench, the more powerful **Name Mode**, which provides a host of extra functionality, or the new **Icon Action Mode**, which combines the ease of use of Icon Mode with the power of Name Mode.

As well as the usual ability to select, display, and drag and drop single or groups of files and directories, in Name Mode and Icon Action Mode each Opus 5 Lister has a range of extra features already built-in. These features include :-

- A *status display* showing information on the files in the selected directory.
- A quick access *toolbar* of custom, icon-image style buttons, each having separate actions for the left, middle and right mouse buttons.
- A custom *pop-up command menu* which by default provides instant access to the main internal Opus 5 AmigaDOS commands such as copy, delete, etc.
- A *pop-up menu* giving instant access to directory functions and to the history of previously seen directories.

- A *pop-up menu* to switch the Lister's status quickly between source or destination, between icon and file display, to lock the Lister's status as desired, and other functions.
- A right mouse button *pop-up menu* which provides quick access to basic Lister GUI functions such as iconify, snapshot, and Lister modes.
- In Name Mode only, the ability to *customise the file display* by field, type, date, file type and other features. You can also customise the font and colours used to display the different types of files.
- In Icon Mode and Icon Action Mode the ability to display a background pattern in the Lister Window.

Custom Button Banks

Just like earlier versions of Directory Opus, Directory Opus 5.5 provides you with the ability to create a bank of custom buttons giving separate functionality to left, right, and middle mouse button clicks. New for Opus 5.5 is the added ability to incorporate a pop-up menu into any button. This allows you to group a larger number of related functions in one place and select the specific one required. Holding down the LMB on a button displays the full list of commands attached to this button. Releasing the LMB over a command brings this command to the top and makes it the default LMB command.

As you may now have come to expect with Opus 5, a Button Bank is not just limited to the traditional old style, static group of buttons. With Opus 5, you can have as many Button Banks as you wish. The banks can be of any dimensions which fit on the screen, and the buttons can show either a text string or a graphical icon-style image. With Opus 5.5 you can customise the look of the button banks and even select from a traditional Amiga style window with full borders and scroll bars or a minimal button display with horizontal or vertical drag bar.

Each button has the potential of executing an unlimited set of instructions made up of any mix of AmigaDOS, Workbench, ARexx, Script or internal Opus 5 commands. Additionally, each button may have a separate set of actions invoked by a left, middle or right mouse click, or the selection of an item from a pop-up menu.

Again, as a result of the object design concept of Opus 5, the custom buttons used in button banks are a special class of an internal Opus Button Object. The same class of object is used for the Lister toolbars, Lister Menus, custom User Menus, Scripts, Hotkeys, and Filetype actions. This means that not only are custom buttons fully interchangeable between different banks, but buttons are fully interchangeable with toolbars buttons, menus, hot keys, etc. So, as well as being able to edit multiple buttons from different banks at the same time, you can even drag and drop 'buttons' between button banks, Lister toolbars and custom menus!

Further, since the editing of button banks is independent of other program operations, you can open, edit and save button banks at any time. The extra flexibility provided by this approach greatly enhances your productivity when creating your own buttons or when editing existing sets.

User Defined Menus

You are not limited to custom buttons. Opus 5 also provides several custom menu systems for you to use; the User Menus are traditional Amiga style global menus available from the Main Window title bar; the Lister Command Menu is a special set of menus available from each file Lister; individual icon pop-up menus may be customised via the Filetype system; and Buttons within Button Banks can each have a pop-up menu of its own.

Scripts

Opus 5.5 has the added ability to execute a specific function or series of command scripts on certain events and actions. These include starting and shutting down Opus, inserting disks, clicking mouse buttons, opening Listers and many other actions. For instance, whenever you insert a disk into the floppy drive, you can easily tell Opus to play a sound file, open a new Lister and display the disk contents.

Hot Keys

Although you have always been able to attach a specific hot key to any button or menu within Opus, Opus 5.5 has added a specific set of user defined hot keys. The keys may be solely for use while Opus is the active program or may be used as global system hot keys. As with all other Opus functions, you may attach a command or custom script to each hot key.

Extended Power

Through these menus, scripts and hot keys, Opus 5 provides the user with the power to extend the basic operations of the program. Each action is fully user definable and may be made to execute a simple command or an unlimited set of instructions made up of any mix of AmigaDOS, Workbench, ARexx, Script or internal Opus 5 commands.

Configurations Settings

Because Directory Opus 5 follows the Amiga Style Guide recommendations as closely as possible, we have been able to rationalise some of the excessive and now redundant configuration options presented by some other programs.

However, Opus 5 still provides extensive user control over the essential elements of the visual environment, plus gives control over the operational behaviour of the Opus 5 command functions.

These settings are grouped into two independent requesters which separately control the *Environment* and procedural *Options*.

From the *Environment* section you can save the complete layout of a particular visual display. This includes not only the screen mode and colour selections but also encompasses all the Listers and button banks, their paths and screen positions. It is very easy to tailor a custom display for a specific job.

Special Paths

Directory Opus has always provided you with shortcuts to access specific paths within your system. With Opus 5.5 we have improved the method and now provide you with a powerful system to customise the visual display of any specific directory in your Amiga. You can now define the Lister position, size and display format for any directory and save these settings to be used whenever you access this directory. You can also easily allocate a specific hot key to bring up any specific directory on demand.

Automatic Recognition of Files

A very versatile feature of Directory Opus 5 is the ability to recognise files by type using a system called *Filetypes*. With Filetypes, you can configure Opus 5 to play animations when they are double-clicked, to load a database program when you double-click on a database file, or to uncompress an archived file when you drag and drop it to a new directory.

Opus 5 comes with several pre-defined Filetypes for many of the common types of files you may encounter on the Amiga. We also provide a full Filetype editor where you may edit current Filetypes and associated actions. It is relatively easy to teach Opus 5 to recognise new types of files and provide commands to be executed when you double-click, drag and drop, or apply special Opus 5 functions to such files.

New for Opus 5.5 is the '*Automatic Filetype Creator*' which will analyse unknown file formats for you and make the creation of filetypes easier and faster.

Drag and Drop!

One of the many powerful features built in to Opus 5 is the extensive use of the concept of 'drag and drop'. This is where you highlight one or more items with the mouse, pick them up on the mouse pointer, drag them to a new place, then release the mouse button. When using Opus 5, if you are in doubt about how to change or edit a button or function, try drag and drop first. You will be pleasantly surprised by how easy it is to move objects and edit buttons and commands using this concept.

A few specific examples of drag and drop are:-

- If the "Select Destination" requester appears you can drag and drop the icon of a destination drive onto the window to set the path.
- Within Function Editors, use drag and drop either to swap function lines around within the one editor or to copy function lines to another editor. If you hold down shift while you drag a function line from one editor to another, the entire function will be copied.
- Add files to a Program Group by simply dropping the program icons onto the Program Group icon.
- Within the Menu Editors, you can simply drag and drop menu items to move them to the required place.

Remember, when in doubt, try drag and drop!

Pop-up Menus

Opus 5.5 adds many new *pop-up "sticky" menus*. Just press the right mouse button over an icon or Lister title bar and you have instant access to the most common functions. If enabled, these can also be accessed in a Name mode Listers by clicking the right mouse button over a file. The Opus Filetype system allows you to add custom entries to these menus to be displayed according to the type of file or icon selected.

Built-in FTP Support

Because of the object orientated nature of Opus 5's design, in Opus 5.5 we have been able to readily integrate a new FTP module as part of the Opus system. Providing you have an active Internet connection, you can now directly access directories on a remote Internet site just as if they were on your local system. Most Opus commands will function invisibly on the remote directory. For example, not only can you copy files to and from a remote site but you can even double-click on a remote picture file to view it through the Opus viewer.



Chapter Three

Introduction to File Management

The theory behind a directory utility such as Directory Opus 5 is quite simple. Instead of having to struggle with a primitive Command Line Interface and 'mysterious' AmigaDOS commands, you are presented with an easy-to-use interface which shows the contents of multiple directories, and presents the various commands in a manner which makes them much easier to use. On the screen, you open up one or more 'directory windows' or file Listers. Into a Lister you can read and display the contents of a directory from any device or volume accessible by the Amiga. You select files and/or directories, then manipulate them almost any way you like. Selected entries can be copied to other Listers, deleted or renamed; text files can be read, picture files can be viewed, and sound files can be heard. Directory Opus 5 offers much more than these "barebones" features, and you will learn more later in this manual.

Files and Directories

The Amiga's DOS (Disk Operating System) deals with two kinds of data arrangement:- files and directories.

Each file and directory must be given a unique name; within a directory you cannot have two files, two directories, or a file and a directory with the same name.

Files

Any data you record on a disk is stored in a file. Files contain information, which may be from a database, from a word processor, from a painting program, or the entire contents of a program.

The size of a file is expressed in bytes, each byte being equivalent to one character. Storing the string "Hello" in a file would use five bytes, since the word "Hello" is five characters long.

Whether a file can be displayed, executed, deleted, edited, or considered as a script file, depends upon its **attributes**.

All files have a **datestamp** which shows the system time and date when the file was last written to.

Files may also have a **comment** of up to 79 characters attached to them.

Directories

To store information in a logical manner, files on disks are generally organised into **directories**, which are often referred to as drawers. If you picture a disk as a filing cabinet, with your programs, database files and pictures as the actual files, then the directories are the drawers of the filing cabinet. Some of these drawers have further drawers inside them, called subdirectories, which themselves contain drawers, and so on, indefinitely.

The directory or subdirectory containing any given subdirectory is known as its **Parent Directory**.

The highest level of organisation is the **Root Directory**. If the directory is a filing cabinet, then the root directory is the room it stands in. The route you take along a directory tree to reach a file is called the **path**. As you proceed along the path, each branch of the tree is separated from the next by a forward slash '/' character.

Chapter Three

For example, on a hard drive named *Work*, the path to the Directory Opus 5 directory should be *Work:Opus5*. To refer to the Directory Opus 5 program, you would use what is called the pathname. This consists of the file's path followed by the name of the file. For example, *Work:Opus5/DirectoryOpus*.

The number of files and subdirectories any given directory can contain is limited only by the amount of space on the disk.

For a more complete explanation of file structure, please consult the Amiga Dos and Users Manuals.

This page is intentionally left blank.



Chapter Four

Installing Directory Opus 5



You cannot run Directory Opus 5 directly from the master distribution disk. It is designed to be run ONLY after being installed on your hard drive.

Installation of Opus 5 is handled by the standard Amiga Installer program and an associated script. Do not attempt to install the program parts yourself. For the correct operation of Opus 5, you **MUST** use the installation procedure provided. Don't worry, the installer script does everything for you, offering you few choices.

Insert the master distribution disk in your floppy disk drive and open the disk icon from your workbench screen. Double-Click on the '*InstallOpus*' icon to start the installation procedure, then follow the instructions on the screen to install Directory Opus 5. The program, related files and directories will be installed into a new directory on your hard disk called '*Opus5*'.

Once the installer script has done its job, Directory Opus 5 will be automatically run so that you may serialise your installed copy to complete the installation. (See *Serialising Directory Opus 5* on page 25).

Technical Details of Installation

Although there is no need for you to be concerned about the installation process, the following is a brief technical discussion about how Opus 5 expects to be installed.

Directory Opus 5 is designed to run ONLY from its own directory on a hard disk. It also expects to have certain subdirectories installed under the Opus5 directory. Briefly, these are

Environment

Contains the files which define the visual display characteristics for screen mode and palette etc, plus links to the other items in the program configuration such as Button Banks and Menu selections. An Environment file also keeps track of any default Lister you wish to open, including their position, current path and display format. By default, on startup Opus 5 will use *Environment / Default*.

Settings

Contains the files which define most of the user-configurable options and special flags. By default Opus 5 will use *Settings / Default*.

Buttons

Contains files detailing the various button bank and similar definitions. Note that all function definitions are stored as button banks. This means that button banks, Lister toolbars, Lister menus, user menus, hot keys, and script files are all interchangeable. Even graphical and textual button banks may be interchanged, but they may look slightly odd! By default, Opus 5 will look for *Buttons / Toolbar* to use as the Lister toolbar, *Buttons / Lister Menu* to use for the Lister menu, *Buttons / User Menu* for the user menu, *Buttons / Scripts* for the scripts, and *Buttons / Hotkeys* for the hotkeys.

Images

Contains the various image files for toolbar and other button images. All images are stored as IFF ILBM brushes, '.info' files or anim brushes. (The default images are 22 x 14 pixels.) There may be more than one Image directory for different sets of images.

Icons

Contains the default icon files ('.info') used by Opus 5 when creating new drawers and associated files.

Groups

Contains any special *Program Groups* you create.

Filetypes

Filetypes are global to the Opus 5 system and are stored in individual files, rather than being stored in one specific configuration file. Any Filetypes present in this directory will be loaded automatically. Opus 5 uses file notification to keep track of changes you make while it is running. Each physical file stores only one actual Filetype definition.

Storage

Contains unused modules and Filetype definitions.

Modules, Libs, C and System

Contains various files, programs, program modules and libraries to control Opus 5 operations.

Help, Catalogs and ARexx

Help contains the AmigaGuide files for context sensitive Opus 5 on-line help, Catalogs contains the Locale language files for non-english versions, and ARexx contains sample ARexx scripts.

Installation Options

Apart from allowing you to select where to install the Opus5 directory, the installer script offers you the choice of having Directory Opus 5 run automatically when you next boot up your computer. If this is not required, choose *option (d)* and you may then run Directory Opus 5 from the Workbench or from the CLI at a later time.

We recommend that you try option (a)!

You will be presented with choices similar to the following

- a) Install Opus 5 as Workbench replacement
- b) Start Opus 5 on boot (own screen)
- c) Start Opus 5 Iconified
- d) Do not start Opus 5 on boot

If you choose *Option (a)*, it will install Opus 5 as a default Workbench replacement, specifically, by replacing your original '**LoadWB**' program in the '**C:**' directory with a new version which will load Opus 5 instead of the usual Workbench program. For safety, the original '**LoadWB**' program is renamed to '**LoadWB_old**'.

Options (b) and (c) will place a small file in your **SYS:WBStartup** drawer which will automatically run Opus 5 when you boot your computer.



If you have installed the 'LoadWB' program from Opus 5 and wish to boot your computer with the old workbench instead, simply hold down the SHIFT key while the computer is booting.

Serialising Directory Opus 5

In the program package you will find a registration card with your personal serial number. Before you can use your new Directory Opus 5 program, it must be personalised with this serial number. When prompted, enter your serial number and other details as shown on the screen. Select '*Register*' when done and Opus 5 will be ready for use.

This would be a good time to complete and return your registration card to the address indicated on the card. Please remember that technical support and upgrades are only available to registered users.

Please safeguard your personal serial number. If you need to re-install Directory Opus 5, you will need to serialise the program again with this number. You will also have to quote this number for technical support and upgrades.



Be sure to enter your registration number EXACTLY as shown on your registration card, including correct upper case and lower case characters as shown.

Running Directory Opus 5

There are several ways of starting Directory Opus 5.

- As 'Workbench Replacement'
- Automatically on Boot
- From the Workbench
- From the CLI

Each of these options is discussed below.

Workbench Replacement (WBR)

This is now the preferred method for operating Directory Opus 5. Running Opus in place of Workbench provides you with only one screen to handle with all the standard functionality of Workbench but with the extra power and ability of Opus. Remember that if you really need it, you can open Opus Listers in Icon mode and operate the system just as with the original Workbench.

As discussed above, when you choose to install Opus 5 to run as Workbench Replacement (**WBR**), we actually install a new **LoadWB** program in your C: directory to handle this job. For safety, we rename your original '**LoadWB**' command to '**loadWB_old**' and replace it with a our '**LoadDB**' program which is renamed to '**LoadWB**' so as to work with a standard Amiga startup-sequence. So, when your startup-sequence makes the call to **LoadWB** at the end of the script, Opus will be run instead of the normal Workbench program.

When Opus is run from boot in WBR mode, all that really happens is that the Opus Main Window is opened on the system's default 'Workbench' screen and the standard Amiga 'Workbench' program is not run. If you are not running Opus as WBR, we suggest that to avoid confusion you run Opus on its own screen and not on the Workbench screen.

Potential Problems with WBR mode.

After the release of Opus 5, we found that some PD utility programs would fail to operate correctly if the 'Workbench' program was not found. In other cases, programs would not find the correct system path list or would suffer from other related problems. After extensive testing of Opus 5.5, we believe that all these problems have now been overcome. However, there are limits. We have seen some PD utility programs which attempt to illegally patch some specific functions of the Workbench program in an unsupported manner. These will not work. If you have a problem, please remove all such utility programs from your startup-sequence, user-startup and the WBStartup drawer, boot with Opus 5, then replace the utilities one by one to find out which one causes the problem.

When run as WBR (via LoadDB), Opus will attempt to use the environment file named '*workbench*'. (If run as a stand alone program, the configuration named '*default*' will be used unless overridden by a command line argument or Tool Type. When running as WBR, these options are not available.)

Also, when run as WBR, Opus will automatically run any programs in your *SYS:WBStartup* drawer to provide compatibility with Workbench.

If required, you may add *Command Line Arguments* to the LoadWB (LoadDB) program line in your startup-sequence. (See below.)

Remember, if you have installed Opus 5 as Workbench replacement and wish to boot your computer with the original Workbench instead, simply hold down the SHIFT key while the computer is booting.



If you ever wish to stop running Opus as Workbench replacement, simply rename LoadWB to LoadDB and rename LoadWB_old to LoadWB.

Automatic Startup on Boot

The modern method of starting a program on boot is to put an icon in the *WBStartup* drawer of your boot disk. You could place the whole Directory Opus 5 program (and icon) in there, but this is a severe waste of space. We have provided a special icon named '*Opus5_Startup*' specifically for this purpose. If not already installed by the *InstallOpus* script, you will find this icon in your Opus5 drawer in the subdirectory *WBStartup*. To start Opus 5 when you boot your computer, simply drag this icon into the *WBStartup* drawer of your boot partition on your hard disk.

The older method of starting a program from boot was to add lines to your startup-sequence or user-startup files. We recommend that you use the *WBStartup* options as above instead of modifying these files.

You can modify the startup behaviour of Opus 5 by changing the Tool Types in *Opus5_Startup* icon. (See page 29.)

Starting From Workbench

If you have not installed Opus as WBR, the quick way to run Directory Opus 5 is from the Workbench. Double-click on the Directory Opus 5 icon and, a few seconds later, the program will appear. It will load the *default* Environment settings and appear either on its own screen or on the Workbench screen, as defined in the Environment. You can modify the startup behaviour of Opus 5 by changing the Tool Types in the program icon. (See page 29.)

Starting From the CLI

To run Directory Opus 5 from the CLI enter:-

DOpus5:DirectoryOpus

This assumes that you have installed the program on your hard drive using the *InstallOpus* script provided and that the correct assignment of '**DOpus5:**' has been made. (The *InstallOpus* script

should have done this for you.) The more technically inclined may notice that there is no RUN command on the instruction line. Directory Opus 5 detaches itself from the CLI, leaving the CLI free to be used for other commands and ultimately allowing you to close the CLI window completely.

Directory Opus 5 can take a series of extra command line arguments to modify the initial startup behaviour.

Tool Types, Command Line Arguments

By changing the Tool Types in either the program icon or the Opus5_Startup icon, or by using command line arguments directly, you can modify the initial startup behaviour of the program. The Tool Types and Command Line Arguments have the following meaning :-

ENVIRONMENT=filename

The filename is the pathname of a special Environment file to be loaded and used. (**Note: This cannot be used with LoadDB in WBR mode.**)

CX_POPUP=yes/no

This defines whether Opus 5 will open its main screen on startup or not. To start Opus 5 iconified, set this to **No**.



CX_POPKEY is now obsolete. The default hot key is still defined as Ctrl - left shift - left Alt but may now be defined directly in the Options / Hide Method. (See page 87.)

QUIET

Tells Opus 5 not to display the 'Loading program...' requester.

Installation

STARTUP_PIC=file

Specifies the path to a picture to be displayed while Opus 5 is loading. This picture can be a standard IFF ILBM, or if you are using Workbench 3.0 or higher, any picture that you have a datatype installed for.

WBSTARTUP=yes/no

Specifies whether Opus should execute the programs in your SYS:WBStartup drawer.



ONLY use this if you are running Opus in WBR mode by executing the main program itself (DOpus5:DirectoryOpus). When Opus is installed normally as WBR and run via LoadDB, this is automatically turned on.

When starting Opus 5 from the CLI, the above arguments may also be used as command line switches.

When running Opus 5 in WBR mode using LoadDB, all the above command line arguments may be used except ENVIRONMENT. In WBR, ENVIRONMENT is always set to read the DOpus5:environment/workbench settings.



Chapter Five

Using Directory Opus 5

This section gives you a short tour of Directory Opus 5. It describes the operation of the program when using its default configuration. Other configuration options are described in later chapters.

Aborting Operations

Before using the program, it is important to know how to abort an action. Most functions can be aborted once they have begun. This can be quite useful if you accidentally start deleting the contents of your hard drive.

To abort an operation, select the '*Abort*' gadget shown in the file Lister. Do not be alarmed if the action does not abort immediately; some functions such as copy, may have to finish with the file they are working upon before exiting.

Because Opus 5 uses extensive multitasking, it will often internally spawn another task or program to perform your selected action. Or, the selected action may be spread over a number of Opus 5's internal tasks. For example, when de-archiving, Opus 5 will spawn your selected archiver, for example, LHA, with the arguments to perform that job. After this Opus 5 will return to monitoring its main screen. To abort such actions, it will be the LHA task you must abort not Opus 5.

Opus Context Sensitive Help



Directory Opus 5 provides extensive context-sensitive help. Simply move the mouse pointer over an object in the current window and press the **Help** key. This will lookup the object in the Opus 5 AmigaGuide-based help system and display the details of the object, button, or command pointed to by the mouse pointer.

If the mouse pointer is over an Opus 5 Custom Button, the help system will display information on the first command attached to that button. Of course, Opus 5 can only show details of its own internal commands. It cannot show details for script files or calls to AmigaDOS programs.



Directory Opus 5 Components

As discussed in the introduction, Directory Opus 5 consists of a number of inter-related objects, each running as its own task or process but communicating with the other objects as required.

The main objects of the Directory Opus 5 system are a base window, called the **Opus 5 Main Window**, plus any number of the following objects:- a window displaying files and directories, called a **Lister**; a window displaying custom buttons, called a **Button Bank**; secondary requesters for functions such as DiskCopy and Format; and from time to time, various windows used to edit and adjust the configuration settings or other operational parameters of the Opus 5 system.

This may all seem a little complicated at first, but in practice it is not! It just uses the capabilities of the Amiga to allow you to do several things at once, if you want. In reality you will find the power and flexibility easy to master.

The Opus 5 Main Window

When Opus 5 is started, it first opens its main display window. This is the handle by which you initially access basic Opus 5 functions.

This window, and its underlying process, provides similar functionality to that of Workbench. As discussed on page 26, with Opus 5 you no longer even need to run Workbench at all and can use the Opus 5 Main window as a replacement for the standard Workbench window.

Like Workbench, the Opus 5 Main Window shows icons representing the disks and volumes available. Once you have mastered using Opus 5 itself, you may also wish to *leave out* other icons for your favourite files, directories and programs.

You can select to open the Opus 5 Main Window on its own custom public screen, on the Workbench screen, or on any public

screen available in your system. You may also use the Opus 5 Main Window in place of Workbench as discussed elsewhere.

Program Application Icons

Opus 5 can display the ApplIcon images which application programs may open as special handles on Workbench. The display of these may be controlled by a switch in the Environment requester as discussed later. If you turn this option on, Opus 5 will hear about these events from the Amiga OS whenever a program attempts to add these to the Workbench display.

Selecting Icons

To access any icon displayed on the main window, simply double-click on the icon with the left mouse button. This will open the volume or device and read its contents into an Opus 5 file Lister display.

Alternatively, you may use the keyboard to select and activate the icons as follows:-

the **SPACE** key toggles keyboard selection mode,
the **Arrow cursor** keys move the highlight,
the **RETURN** key activates the selected icon.

Leaving Out Icons

If you wish to leave out other icons representing files, programs or directories, you can highlight the icon and select *Leave Out* from the Icons menu, select *Leave Out* from the entry's RMB pop-up menu, or more simply drag and drop the item(s) onto the main window from a file Lister display. Left out icons will display a small arrow image to show that they are left-outs. To put icons away, highlight the icon(s) and select *Put Away* from the Icons menu or the icon's pop-up menu.

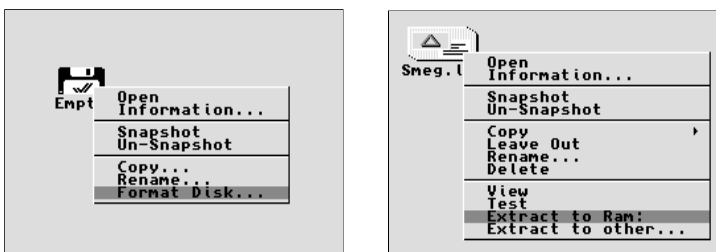


Unlike Workbench, when you 'leave out' files, the icon images are still visible in their original directory.

Icon Pop-up Menus

Each icon displayed on the Opus Main Window has special, *sticky*, pop-up menus, accessed by clicking the right mouse button over the icon. These menus contain a selected set of commands depending on the object to which they are attached, whether Disk, Drawer, Group, or left out program. These pop-up menus are also available on icons in Icon or Icon Action mode Listers and, if activated, are also available on files in Name mode Listers.

In reality, these pop-up menus are **Filetype specific** and are enhanced by the Opus 5.5 Filetype recognition system. From Filetypes you may easily add new menu functions according to the type of file. For example you could add special menus to any LHA file as shown below.



Program Groups

On the Amiga, the concept of *Program Groups* is unique to Opus 5. Instead of having to leave out all your favourite applications on the main window, Opus 5 gives you an easy way to organise your files by allowing you to create custom drawers called *Groups*. In a Group drawer you can collect your favourite applications for immediate access.

As an example, you could create a Group called 'Graphics' and store in it programs such as DPaint, Brilliance, Photogenics or other painting programs. Another example would be to create one or more groups and collect all your related Internet applications in one place. One group called 'Net' could be used to reference applications such as AmiTCP scripts, another called 'WWW' could

hold assorted web browsers, and yet another could hold mailers such as Thor and other Internet utilities.

Instead of having to search all over your hard disk for each program, you could then simply open the 'NET' group, for example, and double-click on the required program to start your Internet connection and access the World Wide Web.

To place applications in a specific Group drawer, open an Opus 5 Lister showing the application's current directory then drag and drop the application icon into the Group window.

For efficiency, Opus does not copy the actual icon or program to the group drawer but only preserves a reference to the actual icon. If you edit an icon in a Group, you are actually editing the original icon. When you double-click on icon in a Group, you are effectively double-clicking on the real icon in its real directory.



Not all icons may be placed into a Group drawer. If a file does not have an associated TOOL or PROJECT icon, it cannot be placed in a Group drawer. Also, you cannot place other drawers in a Group drawer.

Menus

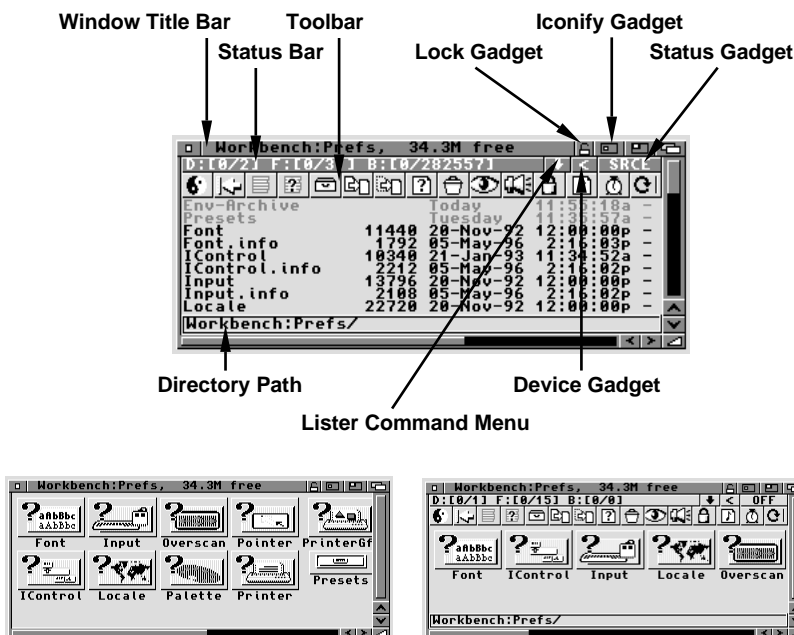
Access to other main Opus 5 functions is provided by global menus attached to this window. (See The *Global Menu* on page 57.)

The Opus 5 File Listers

The hub of the Opus 5 system are the file Listers. Each file Lister is an independent process with its own output window. You may have as many Lister windows open at any one time as you desire. Often, you will select two or more windows to act in concert to, say, copy files between one directory and another. Alternatively, you may only require one Lister to view files in a directory and play a series of sound files.

Many of the features of a Lister display such as status and file colours, display formats and other features can be customised. See the section under the Environment menu for Lister Display and Lister Format.

Opus 5 Listers can display files in one of three ways, in *Name Mode*, *Icon Mode*, or *Icon Action Mode*.



Lister - Name Mode Display

We shall discuss *Name Mode* first since this was the original native display mode of the Directory Opus Listers. This mode is designed to show the list of files and directories in greater detail, as well as providing many extra commands to make file manipulation easier.

In Name Mode, the Lister display window has the following features:-

Lister Window Title Bar

The title bar is at the very top of the Lister display. It is used to display various status or error messages. Usually, this will show the disk volume name, the name of the current directory, and the amount of free space on this drive.

As with normal Amiga windows, if the Lister window is active, the window title bar will be highlighted.

If you are unsure what Directory Opus 5 is doing, it is a good idea to look in the title bar for a hint.

As well as a normal depth gadget, the window title bar also contains (from right to left) a zoom gadget, iconify gadget and display format lock gadget.

Lister Status Bar



Immediately underneath the title bar, is the Lister status display with three embedded gadgets. This area shows details about the current directory and selected items. It also defines whether a Lister is the source or destination for file operations. From the *Environment/Lister Display* menu, you can easily set different highlight colours for the status bars of source and destination Listers. This gives you an instant visual clue to the status of each Lister.

Chapter Five

The Lister status bar has four parts which are described in detail over the following pages.

a) The Lister Message area

The large area on the left shows either a status or warning message about the Lister display, or, more usually, when a device or directory has been selected, it shows information about the directories, files and byte size in the current Lister display. The number of directories, files and bytes is shown in the following format:-

D:[xxx/yyy] F:[xxx/yyy] B:[xxx/yyy] *

where yyy is the total number of items, xxx is the number of each item currently selected, and the asterisk is shown whenever files are hidden by a filter, hide bit, or the filter icons flag. If desired you can customise this display with the setting in *Environment/Lister Display/Status Text*.

b) The Lister Command Menu

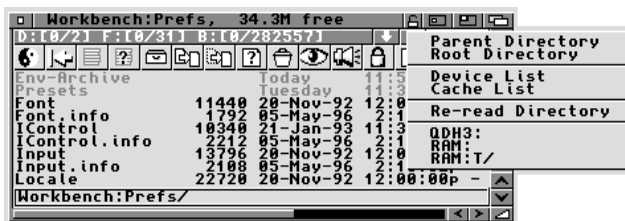
This special gadget provides a pop-up menu which may be used to hold your favourite Opus 5 commands. These commands act only on the Lister to which they are attached.



Using Directory Opus 5

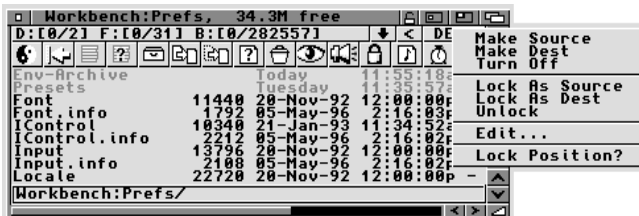
Opus 5 comes with a pre-defined set of commands for this menu. By default, Opus 5 will load the file *Buttons/Lister Menu*, but you may readily customise these defaults to your own requirements. (See *The Menu Editor* on page 174.)

c) The Lister Device Gadget



This special gadget provides a pop-up menu with quick access to the current parent and root directories, the system device list and Opus 5's internal cache list, and the history of previous directories seen by this Lister.

d) The Lister Status Gadget



This special gadget displays the current status of the Lister. It also provides access to a pop-up menu to change the Lister status and display.

Each Lister may be temporarily defined as a source (SRCE) or destination (DEST) for file operations. When activated, a Lister usually becomes the source, and the previous source (if any) becomes the destination. If more than two Listers are displayed at

one time, when activated, the Listers will cycle between SRCE, DEST and OFF in turn.

Alternatively, a Lister may be permanently locked as a source or destination. Listers may also be OFF, that is neither SRCE or DEST.

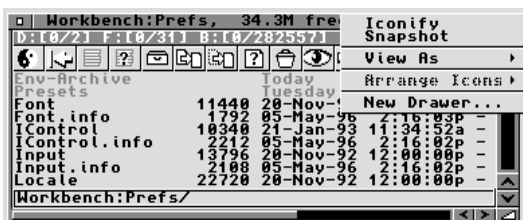
Other options available from the pop-up menu include:-

Edit: Allows you to edit the list format display options for a specific Lister. (See *Lister Format* on page 43.) This may also be accessed from a double-click of the right mouse in the Lister window, depending on the settings in *Environment / Lister Options*.

Lock Position: Normally, each Lister is displayed in a standard Amiga window, which may be dragged to any position and may also be resized. However, Opus 5 provides the option to lock the Lister window in a set position at a set size. A useful option if you wish to mimic other directory utilities such as Directory Opus 4.

Lister Pop-up Menus

Opus 5.5 adds new pop-up menus to Listers. The **Lister pop-up** is accessed from any Lister title bar by clicking the right mouse button. In Icon or Icon Action Mode, the sticky menus may also be assessed by clicking the RMB anywhere over the body of the Lister.



The Lister pop-up menus provide rapid access to common functions including:-

Iconify: Reduces the window to a small icon on the main window. This saves memory but retains the contents of the Lister directory. To un-iconify a Lister, simply double-click on the icon. You can also iconify the Lister from the iconify gadget in the Lister title bar.

Snapshot: Saves the current size, position and display mode for this specific directory path. Next time you open this path it will open exactly as you have saved it. (Note: See the *Environment/Lister Options* section for more details on snapshot.)

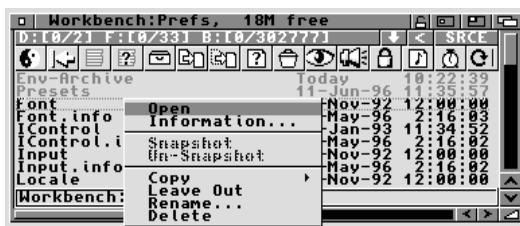
View As: Changes the Lister display to Name, Icon or Icon Action mode. **Show All** displays all files which do not have their own icon by using Filetype icons if they exist, or, if not, by using the default icons as defined by the system default icons in ENV:sys. (e.g. def_project.info etc)

Arrange Icons: When icons are displayed, this arranges the icons in Name, (file) Type, Size or Date order.

New Drawer: Provides a shortcut to creating a new drawer or directory.

Filetype Pop-up Menus

These **Filetype specific** menus are accessed by clicking the RMB over any entry in the Lister, whether directory, file, or icon. These are the same menus as those available from icons on the Opus Main Window. (See *Environment/Lister Options*)



Lister Position, Display Format and Sort Order

When a new Lister is first opened, it will use the system defaults for size, position, mode as defined in the current Environment, unless you have previously done a snapshot of the size and position (and mode) for the chosen directory path. If the Lister is in Name Mode, the display format and sort order will also be taken from the system defaults unless you have saved a special format for this path. This behaviour can be refined further with settings in the *Environment/Lister Display* and *Environment/Lister Options*.

For Name Mode Listers, if a special path format has been saved for the chosen path the Lister will adopt the display format and sort method you have previously saved unless you freeze the Lister format by using the **lock gadget** in the Lister title bar. If this is selected, any special path formats will be ignored and the current Lister format will be used.

Editing Display and Sort Order

The Lister format editor may be invoked from the Lister status menu (Edit) or by a double-click of the right mouse button in a Lister window if that option is enable in the *Environment Editor Lister Options*. The format of each Lister may be changed independently.

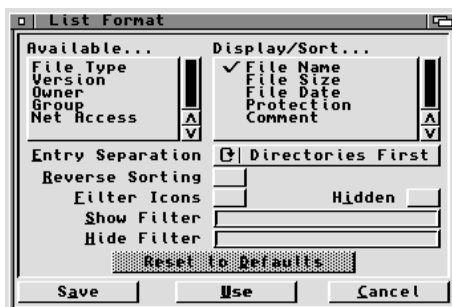
The display format used in each Lister can be made up of any of the following fields, in whatever order you wish:-

File Name	File Size
File Date	Protection
Comment	File Type
Version	Owner
Group	Net Access



The version check understands version format used internally for libraries and devices and also looks for the version string as recommended in the Amiga Style Guide (defined as "\$VER: Name version.revision <(DD/MM/YY)>"). If the version information in the file does not follow the recommended standard, Opus may report anomalous results.

The chosen display order is shown in the Display/Sort list with the field by which the data is sorted tagged with a check mark.



You can move items from the Available list to the Display/Sort list by clicking on them. Alternatively, if you click and hold a given item, you can drag and drop it between the lists, and also drag it to a new position in the Display/Sort list.

Entry Separation: Choose between a display with Directories shown first, files shown first, or a mix of files and directories, sorted in order as selected above.

Reverse Sorting: By default, the display is sorted in alphabetical order, or, if the item starts with a number, then the item is sorted in numerical order. Tick this gadget to display the list in reverse order.

Normally, the Opus 5 file Listers will display all files and directories in the selected directory. By using the filters you can restrict the files displayed.

Filter Icons: By default, all the '.info' icons files are also displayed in the list. Check this gadget if you wish to not display these files.

Hidden: When checked, tells Opus 5 to NOT show any files with the 'H' or hidden bit set.

Show and Hide: These text fields allow you to select a pattern of files to show or files to hide. The pattern may be defined using the standard Amiga wild cards of '#' and '?'. For example, putting '#?.o' in the Hide field, will cause the Lister to hide any file ending in '.o'.

Reset to Defaults: Will reset the format to the current global system default setting. Saving this format will cancel and remove any previously defined special path format.

Save: Ties the defined format to the path displayed in the Lister and permanently saves (snapshots) it for use whenever the defined path is accessed.



Saved formats are remembered forever and always override the system default format unless the Lister lock gadget is closed. Use Reset to Default to clear and forget a stored format.

Use: Accepts this format for this Lister for the current session

Cancel: Abandons all changes.

Dynamic Resorting

Since each Lister supports dynamic resorting, it is easy to change the sorting method in a specific Lister on a temporary or permanent basis. For example, you may wish to see the new files recently added to a directory. To do this, double-click the right mouse button over a Lister to bring up the format editor, select *File Date* as the sort method, tick *Reverse Sorting* and select *Use*. (Alternatively, if Field Titles are being displayed, simply click on the Date field title.)

Special Formats for defined Directories

In some cases, you may often wish to define a specific display format for a particular directory. For example, you may wish normal directories to be sorted in filename order, but with a 'Downloads' directory, you may wish to see the latest files you have received.

As from Opus 5.5, if wish to have a special format associated with a specific directory (such as a Download or Incoming directory), set this directly from the Lister format editor then select *Save*, and the chosen format will be permanently stored for whenever you next access this directory. (See page 93 for more details.)

Lister Field Titles

From the *Environment/Lister Options* settings, you may select to display the names of the fields shown in a Name mode Lister. When these are active, clicking on the name of the field will dynamically resort the Lister, alternatively in ascending or descending order by that selected field.

Lister Toolbar

Immediately beneath the status bar, is an optional toolbar showing small graphic images or icons. Each of these images is actually a normal Opus 5 action button for which you may define separate actions for left, middle and right mouse clicks.

By default, Opus 5 uses the file *Buttons/Toolbar*, but you may easily define your own Toolbar. The images and actions of the Toolbar buttons can be edited from the *Editing the Lister Toolbar* on page 182 for more details. For those who wish to experiment or extensively customise their Lister display, Opus 5.5 has a new internal command, *Set Toolbar*, which allows you to load a new toolbar of your choice into any given Lister.



In Opus 5.5 the Lister window (or button window) no longer needs to be active in order to see right and middle mouse buttons. Clicking RMB or MMB over a Lister toolbar icon will generate the associated RMB or MMB command function without activating the Lister window itself.

Directory Path

At the bottom of the Lister display is a string gadget which contains the full directory path. To go quickly to a specific

directory, you may enter the full path in this box and press return. Opus 5.5 adds a history to the string gadget. Activating the string gadget and pressing cursor up or cursor down will scroll through the previous entries.

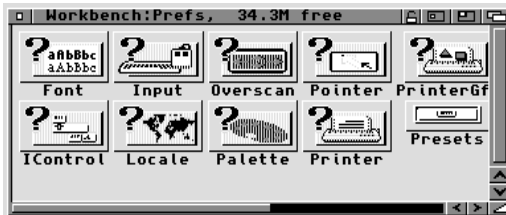
Hidden Parent Button

Each directory window contains a hidden Parent button on the left hand outer edge of the window area. From any Lister in Name or Icon Action modes, when you click the left mouse button on the left-hand window border, Directory Opus 5 will display directory's parent, if it has one. Alternatively, the button second from the left in the default toolbar provides the functions for *Parent* and *Root*.

Lister - Icon Mode Display

Icon Mode provides a simple 100% compatible Workbench style graphical display where files are displayed by their associated icons. It is best used for running PROJECT applications which can often be run more easily from their associated icons, or for quick access to a new disk.

In Icon Mode, the Lister window behaves more like a standard Workbench window. It does not have the status bar and toolbar, nor its associated gadgets, only a standard window display.



Icon Mode is not designed to be used for extensive file manipulation. If you wish to do this, switch to Name Mode. However, you may still easily copy files using the simple drag and drop operation.

The prime purpose for Icon Mode is to provide extra compatibility with a standard Workbench display. In Icon Mode, when you double-click on a PROJECT or TOOL icon, Opus 5 will launch the application as if it had been run from Workbench. Otherwise, the normal Opus 5 rules apply and Opus 5 will look at the file type and follow the instructions you have provided under the particular Filetype definition. (See Filetypes on page 139).

These different modes of operation can be seen by using the master Opus 5 disk as an example. In Name Mode, if you double-click on the *InstallOpus5* icon on the master Opus 5 disk, Opus 5 will call the text viewer and display the script file as text. In Icon Mode, because the *InstallOpus5* is actually a PROJECT icon, a double-click will run the *Installer* program with the *InstallOpus5* script.

In Icon mode, there is no concept of SRC or DEST Listers. Many Opus 5 buttons and command do not work in this mode. Icon mode is designed to emulate the old Workbench display. If you need to have access to Opus functionality, either use Name or Icon Action modes.



Once you switch to Icon Mode, you lose the Lister status bar, toolbar and associated menus. To return to Name Mode or Icon Action Mode, use the RMB pop-up menu or the global menus and select Lister/View As/Name.

Lister - Icon Action Mode Display

Icon Action mode is new for Opus 5.5. It combines the functions of Name mode with an icon display. Since there are SRC and DEST Listers, all the standard Opus internal commands work for Icon Action mode on or between the active Listers, plus you have extended drag and drop control using icons not just filenames.



Both the Icon and Icon Action modes allow you to have a background pattern or picture in the Lister window.

If you *double click* on a drawer or directory icon in either Icon or Icon action modes, a new Lister will be opened using the current display modes unless the directory has a stored snapshot, in which case the stored mode will be used. The position for the new Lister will be the last used position or the snapshot position depending on the settings in *Environment/Lister Options*.

If you hold down the *left ALT key* when you double-click on a directory icon, the parent Lister will be closed automatically when the new Lister opens.

Using the Listers

A Lister is used to display the list of directories and files in a selected directory. You may also display the list of available devices and assigns, or internal Opus 5 cached directories. Custom modules and ARexx handles can also use Listers.

In Name or Icon Action mode, most actions and commands apply only to the highlighted items in the list or highlighted icons, although some commands, such as MakeDir, obviously, act to create a new directory in the current directory as shown in the Lister.

Actions such as Copy, CopyAs, Move etc, act between a Lister defined as the Source (SRCE) and a Lister defined as a Destination (DEST). If you are using a multi-windowed multi-Lister display, be careful that you have set the SRCE and DEST Listers correctly before you attempt a semi-destructive action such as Copy, Move, Delete and so on.

Actions selected from the Toolbar or pop-up menu in a particular Lister act only on that Lister. Actions selected from Button Banks, act on the defined source Lister or between the source and destination Listers depending on the particular command or action.

Using a Mouse with a Lister

Activating a Lister

Single left-click on the Lister window title to activate the Lister window.

Left-click on status bar to make the Lister the source (SRCE).

SHIFT left-click on status bar to make the Lister the destination (DEST).

ALT left-click on status bar turns the Lister OFF.

CTRL left-click on status bar toggles the LOCK option. This toggles a SRCE to locked SRCE! or DEST to locked DEST!.

Depending on the settings in *Environment/Lister Options*, a double right-click over a Lister can bring up the Lister Format Editor.

Moving Around

If there is more information available than will fit into the Lister display, either horizontally or vertically, a limited display will be presented. Use the window's horizontal or vertical scroll bars to reveal the rest of the display.

Alternatively, you can also scroll up and down, and left and right, by holding the right mouse button and moving the mouse. These actions are available in all Listers irrespective of the display type.

Selecting Files and Directories

To select or highlight a file or directory, simply click on it with the left mouse button.

To highlight multiple items, click and hold the left mouse button and drag downwards or upwards to select the adjacent items.

Drag and Drop

Dragging selected items horizontally allows you to pick them up and move them out of the Lister. You can then drop them either in another Lister or on the Opus 5 main window, and even on other objects including the various editors. This is called *Drag and Drop*.

Specifically, to drag and drop, select one or more files and/or directories, and either

move horizontally outside of the window while holding the button down

or

press right button while holding left button. If the Lister already has files selected, use ALT left-click to "pickup" the currently selected files.

Unless otherwise defined, drag and drop of a file copies the item to the new location. For consistency, normal drag and drop operations on a directory perform the same copy function. The default action for drag and drop for all internal Filetypes is to Copy the file, but, this can be overridden by user-defined settings.

Directories

Double-click on a directory to enter and display that directory.

Drag and drop a directory to copy the directory to the new location, or leave it out if dropped onto the main window.

SHIFT drag and drop of a directory does not copy the directory but displays the contents of the directory in the destination Lister. (Actually, it is holding down the SHIFT key when you drop the directory which does the trick!) *SHIFT double-click* on a directory to open up a new Lister for that directory.

SHIFT ALT double-click on a directory to split the current Lister display and open a new directory Lister.



These actions for directories can be further defined by using the special Opus "directory" Filetype. See Filetypes on page 139.)

Double Click Power!

Some of the power of Opus 5 is revealed by a simple double-click on a file. In Opus 5.5 we have extended the double-click system depending on the Lister mode, either File or Icon (or Icon Action) modes.

Name Mode - Double-Click

When you double-click on a file Opus 5 examines it to see if it can identify the type of file. If a file matches a previously user-defined Filetype, for which the appropriate function has been defined, the function is executed.

If a file does not match a user-defined Filetype, it is tested against the internal Filetype definitions, which are as follows:-

Filetype	Function
Executable Program	Run
ILBM picture	Show
Picture (DT)	Show (via datatypes)
ANIM Animation	Show (plays animation)
8SVX sound	Play
Sound (DT)	Play (via datatypes)
Icon	Display IconInfo
Opus 5 Buttons file	LoadButtons
Opus 5 Environment file	LoadEnvironment
Opus 5 Options file	LoadOptions

If none of these match, the action taken depends on the setting of *Options/Miscellaneous/Filetype Sniffer*. Opus 5.5 provides a new **Automatic Filetype Creator** to assist you in creating and testing file types. If the *Filetype sniffer* option is selected, Opus can examine ("sniff") the file in greater detail and install or create a new Filetype for this type of file. (See Filetypes on page 139.)



Otherwise, the **SmartRead** function will be called. This will display the file as ASCII (with or without ANSI sequences) or in hexadecimal, depending on its contents.

👉 ***In a Name Mode Lister, if you wish to check how Opus recognises a specific file, add "File Type" to the display format.***

Name Mode - SHIFT double-click

If you hold down the SHIFT key and double-click on a file, Opus looks to see if the file has an associated icon and if so will perform a double-click on that icon. This allows you to run many programs and installer scripts directly from Name Mode but through the project icons just as if from Workbench.

Icon or Icon Action mode - Double-Click

A double-click on an icon acts the same as Workbench, running the project or tool as defined in the icon.

A SHIFT double-click in Icon or Icon Action mode has no special significance unless more than one icon has been highlighted (by holding down the SHIFT key and highlighting multiple icons or drag selecting icons). In this case, a SHIFT double-click will act the same as Workbench and search the list of highlighted files to find the first TOOL, run this program and feed to it the list of the other PROJECT files.

Clipboard Cut and Paste

Opus 5.5 now fully supports the Amiga clipboard. You can now copy the displayed text from any string gadget such as the Lister

Directory Path using ***right-Amiga-C*** and paste this into another gadget (or any other program which supports the clipboard) using ***right-Amiga-V***. Remember all Opus string gadgets support copy and paste.

Support is also provided directly in Name Mode Listers. You can directly copy the list of highlighted files to the clipboard with ***right-Amiga-C***. This copies the full path of the files to the clip. (e.g. work:smeg/filename). If you use ***right-amiga-shift-C***, only the filenames themselves are copied, not the full path name.

Using the Keyboard with a Lister

When a Name Mode Lister is active, the cursor keys or keypad cursor keys may be used to scroll the Lister display up, down or left or right. Pressing **SPACE** toggles a special keyboard selection mode. In Name Mode, the cursor keys may be used to move the highlight up and down the list and the following sequences may be used. In Icon modes, some of these key sequences will selectively highlight icons in the Lister, others do not apply.

Key	Function
Home / CTRL cursor up	top of list
End / CTRL cursor down	bottom of list
Pg Up / SHIFT cursor up	up one page
Pg Dn / SHIFT cursor down	down one page

Other key sequences include:-

- **TAB** activates the next (non-busy) Lister.
- **SHIFT TAB** activates the previous (non-busy) Lister.
- **RETURN** (in keyboard selection mode) toggles selection status of a file. In non-keyboard mode, it activates the path field at the bottom of the Lister.
- **Enter** (in keyboard selection mode) performs the same action as a double mouse click.
- **SHIFT-ESCAPE** closes the Lister.

Other key functions include:-

Del	Format Editor
/ or BACKSPACE	Parent ***
: or ;	Root ***
*	Select
(All
)	None
-	Toggle
+	DeviceList

(*** plus SHIFT to open a new Lister)

Quick Access to a File Name in Name Mode

To scroll quickly to the first file entry starting with a particular letter, press that letter; the list will scroll as close as it can get to the first entry. Press a letter in conjunction with either SHIFT key to scroll to the first Directory beginning with that letter.

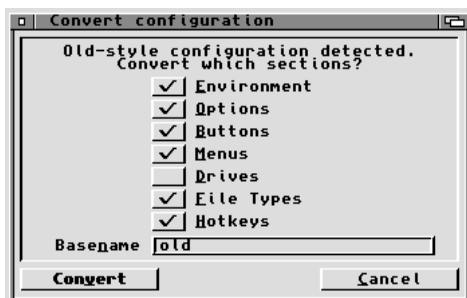
Extended Key Selection Gadget

For those large directories and CDROMs where more than one key is required to choose a specific file, Opus 5.5 adds an *extended key selection gadget*. If enabled from the *Options/Miscellaneous* setting, this provides a pop-up string gadget which allows you to enter several characters to match a file. Again, using capital letters scrolls to directories instead of files.

Converting Directory Opus 4 Configuration Files

Directory Opus 5 is able to detect and convert the old configuration files from Directory Opus 4 ONLY. Opus 5 is not able to convert environment files from earlier versions. If you have a version earlier than version 4 then you will have to convert the files using version 4 first!

If you select *Environment/Open* in Opus 5, and try to load an old version 4 configuration file, you will be asked which parts of the old configuration you wish to convert. The basename you specify will be used when creating individual files corresponding to Button Banks, etc.





Chapter Six

The Global Main Menus

Many of the main functions of Directory Opus 5 can be accessed from the menus from the Opus 5 Main Window. These are known as the global Menus and may be accessed in the usual manner with the right mouse button from the main window.



The right mouse button is used for other functions in Opus 5, depending on the actual position of the mouse. For example, when the mouse is over a file Lister, Button Bank or an icon, pressing the right mouse button will not access the global menus. Instead, it may attempt to activate the function attached to the button or may activate one of the special pop-up menus.

Remember that the file Lister status bar provides its own pop-up menus as well which are accessed with the LEFT mouse button. With Opus 5.5, the object (Lister or button bank) no longer needs to be active to see button events. All are passed through to the underlying object whether it is active or not.

Directory Opus 5 global Menu functions are

The Opus Menu



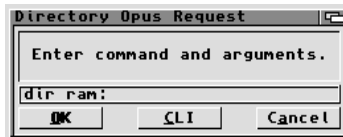
Backdrop

Converts the Main Window to a special borderless window that is always behind all other windows on the screen. This setting is saved with the Environment.



Execute Command

Allows you to start an AmigaDOS command without having to open a new Shell. Opus 5 will open a requester for you to enter the command and any arguments.



For Opus 5.5, this requester now has a history and keeps track of the last 20 commands that you have entered. You can access these by pressing the up and down arrow keys. It also provides access to the internal Opus CLI for advanced users.

If required, Opus 5 will open a new CONSOLE window to output the results of the command. The window will remain open until you select the close gadget.

For shell and AmigaDOS commands, the current directory for the Execute Command is RAM:



About

Displays information about Directory Opus 5 including the version number and your registration details. Try clicking on the animation with the SHIFT key held down.



Hide

Iconifies Directory Opus 5. This command closes the Directory Opus 5 window and screen, deallocates as much memory as possible, and then opens a small window on the Workbench screen. This is known as iconifying, and allows you to have Directory Opus 5 constantly available, while using the minimum amount of memory.

To determine the initial position of the iconified window, especially if you start up Directory Opus 5 in the iconified state, position the iconified window where you would like it to appear, then un-iconify and re-enter Directory Opus 5, and save the Environment. (See *Environment Editor* on page 73.)

To re-enter Directory Opus 5, simply activate the iconified window with a click of the left mouse button in the window, then press the right mouse button. If you wish to quit Directory Opus 5 without going back into it, just click the Close button at the far left of the iconified window.

Iconify Options

There are several options which allow you to define the iconify method you prefer. For more details see the *Settings / Options / Hide Method* menu.

If you have the Hide Method set to *AppIcon* (or *Clock*), you can drag icons from Workbench and drop them on the Directory Opus 5 icon (or clock window). This will have the same effect as if you displayed the file in a Directory Opus 5 window and double-clicked on it. For example, if you drop a picture on the window, it could call the **Show** function.

If you have the Hide Method set to *HotKey Only*, the only way to re-enter Directory Opus 5 is by using the defined hot key combination. This defaults to **Ctrl-left Shift-left Alt** (the Ctrl key, the left Shift key and the left Alt key held down simultaneously). This may be changed directly from *Options / Hide Method*.



Quit

Shuts down all open windows and quits Directory Opus 5. If Opus 5 has launched any associated tasks on its main window, you will have to quit any such programs before Opus 5 can fully close down.

The Lister Menu



New

Opens up a new blank Lister relative to the mouse position. When the new Lister opens it can be told to display the Device List. Either select the desired volume or press RETURN to activate the path gadget where you can manually enter a path.



Open Parent

Opens a new Lister and displays the parent directory of the current SRC Lister.



Close

Closes the current active Lister. Note that no warning is given, the currently active Lister will close immediately.



Close All

Closes all open Listers.



Make Source

Make Dest

Turn Off

Lock as Source

Lock as Destination Unlock

Unlock



These menu selections apply to the currently active Lister while it is in File or Icon Action mode. The actions are identical to those chosen from the Lister ToolBar pop-up menu. See Lister Toolbar for more details.



Unlock All

Unlocks the SRCE or DEST status of all Listers on the screen. If multiple SRCE or DEST Listers are present, they will be changed to SRCE, DEST or OFF depending on the order in which they were invoked.



Edit

Displays the *Format Editor* which allows you to change the display format of the currently active Lister's display. You may also invoke the Format Editor from the Lister Toolbar pop-up menus or by double-clicking the right mouse button on a Lister display area.

(See *Lister Display Format* on page 61.).



Edit Lister Toolbar

Displays the *Button Editor* which allows you to customise the icon images and commands used in the Lister Toolbar.

The Editor may also be invoked by holding down the ALT key and clicking on one of the toolbar icon images.

(See *Editing the Toolbar* on page 182.)



Edit Lister Menu

Displays the *Menu Editor* which allows you to customise the user pop-up menus in the toolbar. (See the *Menu Editor* on page 174.)



Tile - Horizontally or Vertically

Arranges the displayed Listers to fit equally within the Opus 5 Main Window with either horizontal or vertical priority. If the main window is in Backdrop mode, this will tile the Listers equally over the whole screen.

The tile function provides some easy methods of setting up a multi-windowed display. For example, if you have button

banks on the screen, set the main window to non-backdrop and arrange its position and size to just inside the borders of your button windows. Then, tile the Lister in the required fashion. This will neatly arrange the Lister and your button banks.



Cascade

Cascades the displayed Lister within the borders of the Opus 5 Main Window. If the main window is set as a backdrop, this will cascade the current Lister over the full screen.



Snapshot

Snapshots the size, position and mode of the currently active Lister. With Opus 5.5, this information is now longer stored in the file's .icon file but stored internally and on disk for later reference. This snapshot is for Lister only and should not be used to snapshot positions of icons. (See the *Icons/Snapshot* menu on page 65.)



Un-Snapshot

Removes any snapshotted information about size, position and display mode for the currently active Lister. With no snapshot, the Lister will open using the system default size, position, and mode.



View As

Selects the type of display to be shown in the current Lister, either a Workbench style with icons or a more traditional Opus filename mode.

Name: Selects the normal Opus 5 file display which shows extended filename and associated details. This is the default mode for Opus 5 file management operation. In Name mode, the Lister display has the extra features provided by the Status Bar, Toolbar and pop-up menus.

Icon: Selects to display files by their icons instead of by the extended filenames. Normally only files with associated icons will be displayed.

Icon Action: Selects the new combined mode which shows icons plus the extra features provided by the Status Bar, Toolbar, path gadget and pop-up menus.

Show All: For Icon Mode and Icon Action Mode only, it tells Opus 5 to display all the files and directories using pseudo-icons for those which do not have real icons. These are provided by Filetypes or, if not defined, are chosen from the Amiga defaults as defined in ENV:Sys.

The Icons Menu



New - Drawer or Group

Drawer: Calls the *makedir* command to add a new drawer to the current Lister.

Group: Creates a new Program Group under the name you specify.

Program Groups

A unique feature of Opus 5 is the ability to create Program Groups; pseudo-directories in which you place a reference to commonly used applications. Instead of having to leave out these favourite programs on the main window, or search through multiple directories, you can now simply group them and access them easily by opening that group. For example, you could create a Group called 'Graphics' and store in it programs such as DPaint, Brilliance, Photogenics and so on.

To place an application in a particular Group, open the group window, open an Opus 5 Lister showing the original application directory and drag and drop the application icon into the Group window. (You can also drag and drop the application directly onto the Group icon.)

Not all icons may be placed into a Group drawer. If a file does not have an associated TOOL or PROJECT icon, it cannot be placed in a Group drawer. Also, you cannot place other drawers in a group drawer.

The nature and actions of Group icons has changed significantly from those of the original Opus 5. Originally we provided a special menu to delete program groups and remove programs from a group. In Opus 5.5, these commands have been incorporated into the normal Delete command from the Icons menu or directly from the icon's pop-up menu.

In earlier versions of Opus 5, the icons shown in a Group drawer were actual copies of the real icon. This method led to some problems, all of which have been overcome in Opus 5.5. Now, the original file and associated icon is not copied or moved, instead the icon shown is simply an internal Opus reference to the real icon. These pseudo - icons can be deleted, renamed or moved to another group but are simply a reference to the original icon in its true place on your hard disk. However, be aware that if you access the icon itself by IconInfo or a double-click, then Opus translates and performs this act on the real icon. Double-clicking on a group icon IS a double-click on the real icon. The associated program cannot detect that the double-click came from Opus.



Open

Acts the same as a double-click on an icon. If the icon is a disk or drawer, Opus 5 will open a new Lister and display the directory contents. If the icon is a PROJECT or TOOL, it will examine the associated file to see if it knows the specific Filetype. If the file matches the Filetype definition and an appropriate function has been defined, the function will be executed. If the file does not match a user-defined Filetype, it is tested against the internal Filetype definitions and action is taken accordingly. (See Filetypes on page 139.)



Information

Displays status information about the selected icon. It also allows you to edit the Tool Types and other information in an application icon.



Snapshot

Saves the current layout position of the selected icon or icons within the currently active window or group, or the layout of the icons within a group window. The next time you run Opus 5, icons which have been snapshotted will appear in the previously saved positions.

Icons: Snapshots the current position of the selected icons in the active window. Only highlighted icons within the active window are stored.

Window: Snapshots the current position of the currently active window. Only the window position is stored, not the position of the icons.

All: Snapshots the full layout of the currently active window including the positions of all the icons within. This function effectively does a *Select All* followed by a *Snapshot Icons* plus *Snapshot Window*.



To snapshot all the icons in a Group window, arrange the window and the icons as you wish (maybe use the menu Icons/CleanUp), then select Icons/Snapshot to save the positions of the icons within the window. Then, select the Group Icon itself and select Icons/SnapShot to preserve the position of the window itself.



Un-SnapShot

Cancels the snapshot position of the selected icon or icons. The next time you run Opus 5, the icons will be repositioned automatically.



Leave Out

Moves the selected file from a Lister onto the Opus 5 Main Window for easy access. To distinguish left out icons, Opus 5.5 adds a small arrow at the bottom left of the image. Files and directories left out in this manner will appear in the Main Window next time you run Opus 5. Note that the actual file is not moved from its original directory, Opus 5 only stores a reference to the file. Files may be temporarily left out on the

Opus 5 main window. To position a file permanently on the Opus 5 window, you must highlight the file and select *Leave Out*.

Unlike Workbench, with Opus 5 any left-out icons are still visible in their original home directories.



Put Away

Removes the left-out highlighted icons from the Opus 5 main window.



Select All

Selects all the icons in the currently active window, either the Opus 5 Main Window or the active Lister. This gives you an easy method of snapshotting all the icon positions at once. Once you have selected all the icons, use *Snapshot* from the Icon menu to save the positions. When you have multiple icons selected, selecting and dragging any icon with the mouse allows you to pick up and drag all the selections as one.



Clean Up

Attempts to adjust the position of all the icons in the currently selected Lister or the Opus 5 Main Window to their optimal positions within the confines of the window dimensions.



Reset

Resets all the icon positions to those currently stored in the icon itself from the last snapshot operation.



Rename

Provides the option to rename the selected icons



Delete

Deletes the highlighted icons (or files) from the currently active Lister.



Format Disk

Displays the Opus 5 *Format Requester* and allows you to format disks. (See the *Format Requester* on page 187 for more details.)



Disk Information

Displays some information about the disk the active directory resides on, including space used and free, datestamp and number of errors on the disk.

The Buttons Menu



New

Creates a new Button Bank for either text or graphic buttons. When first opened, the button bank will have only one button. The size of the button bank and the definitions for each button may be changed by calling the *Button Editor*.



Load

Loads an old Button Bank from disk. The loaded bank will appear on the screen in the position last saved with the button bank, or in the position it was in when you saved the Environment settings.



Save

Saves the selected Button Bank to disk under a given name.



Close

Closes the currently active (selected) Button Bank.



Edit

Displays the *Button Editor* and allows you to edit the definitions of all buttons in all Button Banks currently open. You can readily edit several Button Banks at once. ***While the Button Editor is open, the buttons cannot be used as normal function buttons.***

The Settings Menu



Clock

Toggles the display of a clock in the Opus 5 main window title bar.



Create Icons

When Opus 5 creates a new directory, this option toggles whether Opus 5 will create the associated icon or '.info' file as well.



Default PubScreen

Makes the Opus 5 screen the system's Default Public Screen . Programs which do not open their own screen will generally use this when opening windows. With Opus 5 as the default pubscreen, most Workbench programs, the Shell, Clock and other utilities will now open on the Opus screen instead of the Workbench. However, this behaviour is program dependent.

This setting has no effect when Opus is running on the Workbench screen as Workbench replacement.



Recursive Filter

If the *Recursive Filter* is enabled, Opus 5 will prompt for an optional file pattern when a function operates recursively on files within sub-directories. If you enter a file pattern, only files matching that pattern will be operated upon. For example, you could select a group of directories and delete only files within those directories ending in ".o", enter "#?.o" as the file pattern. With this option turned off, all files within selected sub-directories will be acted upon. The recursive filter applies to the commands *Copy*, *CopyAs*, *Move*, *MoveAs*, *FindFile*, *Search*, *Protect*, *DateStamp* and *Comment*.



Environment

The Environment provides user control over the visual elements of the Directory Opus 5 display. It holds information

on such things as the screen display mode and backdrop pattern options, the colours and default format used globally for Lister displays, the user selected colours and many other items.

The Environment also keeps track of all the other settings used to make up the current display. This includes the current settings for :-

- Options (Menu Settings/Options),
- the position and formats of any default Listers,
- Lister ToolBar definitions,
- Lister user menu definitions,
- global user menu definitions,
- Button Banks currently loaded,
- Scripts and HotKeys.

When an Environment is saved, all current and saved positional information will also be stored in the file if the Save Layout option is checked.



The actual items such as Button Banks, Lister Toolbar , custom menus and so on are not actually stored in the Environment file. Only a reference to the items by their filename is stored in the file itself. When you load a different Environment, Opus 5 sets the display to the new values then attempts to load other elements of the display by reference to their filenames. If, for some reason you rename or delete an item, such as a Button Bank, outside of the Opus 5 Environment system, the next time you load an Environment which references this item, Opus 5 will not be able to find this element.

If no custom Environment has been specifically saved, when Opus 5 is first run, it will attempt to load the DOpus5:Environment file named 'Workbench', if running on the Workbench screen (WBR) or the file named 'default' if running on its own screen. This environment will reference the following as the system defaults:-

- Settings/Default, Buttons/Toolbar, Buttons/Lister Menu,
- Buttons/User Menu, Buttons/Scripts, Buttons/HotKeys
- plus any Filetypes defined in Filetypes.



Environment Menu

Edit: Displays the *Environment Editor* which allows you to change the visual display characteristics used by Opus 5. (See the *Environment Editor* on page 73.)

Load: Loads an Environment file from disk and resets the visual display of Opus 5 to that defined therein.



Load Environment resets the screenmode to that as defined in the stored file. If the stored environment uses a screenmode of Workbench:Clone, the Opus 5 screen will be changed to clone your current Workbench screen. If this is not the same as when you saved this environment, Opus 5 will adjust the positions of Listers, buttons, and left out icons automatically to fit this new (current) screen size.

Save: Saves the selected Environment using the name under which it was loaded. If no Environment had been loaded, Save will save the current Environment under the name 'default' if Opus 5 is running on a custom screen or 'workbench' if using the Workbench screen.

Save As: Saves the current Environment to disk under the name you specify.

Save Layout?: When checked, the current layout of all file Listers and Button Banks on the screen will also be saved along with the environment.



Options

Displays the *Options Editor*. The Options settings provide control over the various operation commands and associated behaviour provided by Directory Opus 5. These functions and commands include Caching, Copy, Delete, Hide Method (Iconify), Icons, Locale and others.. See page 85.



File Types

Displays the *Filetype Manager* which shows the currently known Filetypes and allows you to edit the Filetype definitions, events and actions. (See page 139.)



User Menu

Displays the *Menu Editor* which allows you to edit, delete and define new menus items for the global User menu.



HotKeys

Displays the *HotKey Editor* allowing you to configure the action of any key sequence to perform a range of custom functions. See page 129.



Scripts

Displays the *Script Editor* allowing you to create custom scripts and commands for various events. See page 133.

The Default User Menus

The User menus may be configured to use any of the internal Directory Opus 5 commands or any calls to other functions you may wish to use. (See the *Menu Editor* on page 174 for details.)

Directory Opus 5 comes shipped with a default set of user menus. We strongly urge you look at these supplied menus from the Menu Editor. This will help you learn how to add your own selections.

The actual default user menu shipped with Opus 5 may change from time to time. Some of the default menus may include:-

Format: Calls the internal command *Format*. It displays the Opus 5 *Format Requester* which allows you to easily format new or old disks. (See the *Format Requester* on page 187.)

DiskCopy: Calls the internal command *Diskcopy* which displays the Opus 5 *Disk Copy Requester* so you may make copies of your disks. (See the *Diskcopy Requester* on page 185 for details.)

Device List: Calls the internal command *Devicelist*. It displays the current device list in the current SRCE Lister or opens a new window if required.

Cache List: Calls the internal command *Cachelist*. It displays the current list of cached directories in the currently defined SRCE Lister or opens a new window if required.

LHA Add: LHA View and LHA extract: These menus use AmigaDOS calls to the LHA program to perform the respective functions.



The LHA program itself is not supplied with Directory Opus 5 and must be installed in your system for these options to work correctly. Because they are shareware, archive programs such as ARC, ZOO and LHA are not included on the Directory Opus 5 distribution disk. They are available free on most bulletin boards and PD collections such as AmiNet etc.



Chapter Seven

The Environment Editor

The Environment provides user control over the visual display elements of the Directory Opus 5 display. This includes the Display Mode, Display Options, the layout of the file Listers, Colour Palette and more.

Desktop

This editor lets you change the look and operation of the desktop.



Distinct icon positions: When turned on, all icon snapshots do not affect the main icon position, but instead are stored with the environment file. When turned off, the standard icon position is used.

Don't cache icon images: Icon caching improves the speed of displaying icon mode and icon action mode Listers. However, if you find this causes problems, you may disable it. This setting is only checked at Opus startup - you will need to quit and restart Opus after altering this setting.

Don't remap icons: This setting allows you to stop Opus from remapping eight colour icons to the top and bottom four colours in the palette.

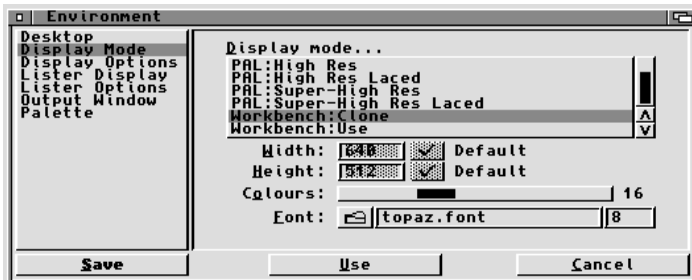


The way the colours of Amiga icons are displayed has become very confusing with a number of competing schemes. These differ from the way the standard Workbench displays the icon colours. You may have some problems with PD utilities such as MagicWB, NewIcons etc depending on the exact way you have implemented your colours and screen depth. With more than eight colours, the pen colours used for the icon's colour can vary depending on the system you have chosen.

Font: Allows you to specify the font used for the names of the icons displayed on the desktop.

Hidden drives: By default, Opus will display an icon on the Main Window representing each of the volumes available on your Amiga. But, you may elect to not display them if you wish. Simply click on the entries you wish to hide and they will not be shown on the Main Window.

Display Mode



The screen Display Mode requester allows you to specify the mode, size and depth of the Directory Opus 5 screen. It displays the list of the available display modes. The modes available will vary depending upon the version of your Amiga Operating System and the monitors currently installed in your system.

The characteristics of the selected display mode are shown at the bottom of the window. There are two special items on the list:-

Workbench:Use Causes Directory Opus 5 to open a window on the Workbench screen and not to open a custom screen. The width and height of this window may be changed, but the number of colours is fixed at the current Workbench depth.

Workbench:Clone Causes Directory Opus 5 to open a screen in the same mode and exactly the same size as the Workbench screen. The size of this screen cannot be changed, but you may modify the number of colours.

Other user settings include :-

Width & Height: These fields allow you to specify size of the custom screen.

Default: When this button is checked, the width or height fields can not be changed and the field will display the default value. Un-check this button to edit the field. When you are using Workbench:Clone, the Default button cannot be deselected.

Colours: Select the screen depth by number of colours.

Font: Select the default screen font for the Opus 5 screen. This is the font that is used for the screen and Lister title bars and for requesters.

Display Options

These options allow you to define whether to use a backdrop pattern or picture on the Opus 5 Screen, and allow you to configure some of the Workbench 'application' functions.



The **Backdrop Pattern** allows you to display backdrop patterns or pictures in the Main Window or Listers (WB3.0+ only). The name of the preferences file is entered into the **Prefs:** string gadget. Many people use the same preferences file as used for Workbench, but you may also have an independent setting for Opus. Opus 5.5 now allows backdrop pictures or patterns on the Main Window as well as in Icon and Icon Action mode Listers.



You cannot just enter the name of a picture file! You must use a standard Amiga 'WBPatternPrefs' preferences file created by the Amiga WBPattern program.

To create a new pattern preferences file, select the '!' gadget on the right of the **Prefs:** string gadget. This will run the Amiga *WBPattern* program where you may define the backdrops you wish to use. The 'workbench' field sets patterns for the Opus Main Window and the 'windows' field sets the Listers windows. Then, 'SaveAs' your preferences under a new name (e.g. dopus.prefs), then specify this file for the Opus backdrop. Note that the *WBPattern* program opens on the Workbench screen, so if you are not running Opus as WBR, you may need to bring Opus to the front after using it.



Backdrop pictures have their colours remapped by default. If you wish the picture not to be remapped, add a '.noremap' suffix to the picture name. The name of the picture itself must have the suffix, not the prefs file. If the picture name ends in a ".exact" suffix, Opus will tell datatypes to remap this picture using PRECISION_EXACT rather than PRECISION_IMAGE.

The Workbench emulation allows Opus 5 to intercept some of the system calls to Workbench to provide the same functionality. The options are :-

Display AppIcons: Tells Opus 5 to display all application icons in the Opus 5 window.

Display Tools Menu: Tells Opus 5 to add any WorkBench application menus to a global 'Tools' menu.

Hide Bad Disks: By default, Opus 5 will show any Non-DOS disk icons (as does Workbench). This option allows you to hide these icons to unclutter your backdrop window.

Lister Display

This allows you to choose the default values for colours, font and format for the Opus 5 Listers.



Lister Elements: Select the item then select the foreground and background colours as desired.

Status text: You can customise the Lister status bar display. Several special sequences beginning with % are available via the button to the right of the string requester. The special sequences are:-

%fs	- Selected file count
%ft	- Total file count
%ds	- Selected directory count
%dt	- Total directory count
%bs	- Selected byte count
%bt	- Total byte count
%h[<text>]%	- Hidden file count/flag

The **%h** sequence can work two ways. If you omit the optional *<text>*, then the number of hidden files will be displayed here. If you supply the *<text>* argument then it will be displayed literally instead of a number whenever any entries are hidden, otherwise the field will be left blank. An entry will be considered hidden if it has been filtered out by the *Show* or *Hide* filters, if it is an icon and *Filter icons* has been enabled, or if it has its H bit set and the *Hidden* switch has been enabled in the Lister format requester.

As an example, this is the default status text:-

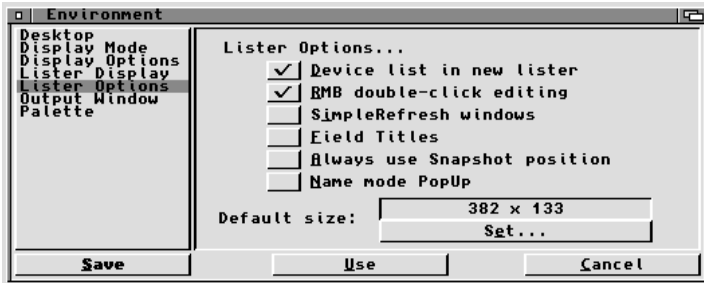
D:[%ds/%dt] F:[%fs/%ft] B:[%bs/%bt] %h*%

Select Font: From a font requester you may select the font and size you wish to be used in the Lister display. The font selected will be used to display all the elements in the Lister display. Only one font can be used in a Lister.

Default Format: From the List Format requester, you may select the default format you wish to be used whenever a new Lister is opened.

Lister Options

These options control the default size and some user actions of the Opus file Lister.



Device list in new lister: When a new Lister is created, Opus 5 will display a *DeviceList* in the window if this setting is enabled. Otherwise it will be blank.

RMB double-click editing: When selected, double-clicking the right mouse button over a Lister will invoke the *List Format Requester*. Otherwise, the right mouse button can be used to scroll the list display horizontally and vertically.

SimpleRefresh windows: This option tells Opus to handle the refreshing of Lister windows rather than let it be done by the system. Enabling this option will mean Opus Listers will use less memory, but may be refreshed somewhat slower.

Field Titles: This option tells Opus to display the name of each display field at the top of the Lister when in Name mode. This setting is global and applies to all Name mode Listers. When field titles are displayed, clicking on the field name will resort the list in ascending or descending order by that field.

Always Use Snapshot Position: When a Lister is closed, Opus makes a temporary snapshot of the size, position and mode for the specific path displayed in the Lister. When you next open this same path, Opus uses this temporary snapshot to re-display the Lister exactly as when you last used it in this

session. However, some users prefer to have Lister always appear in a set size, position and mode. To do this, open a Lister for the required path and permanently snapshot the Lister details from the *Lister/Snapshot Menu*. If this option is set, whenever Opus opens a new Lister of this path, the permanent snapshot will always be used instead of the temporary one.

Name Mode Popup: Enables the RMB *Filetype pop-up menus* in Name mode. These are the same menus as the icon pop-up menus. They provide access to a set of common functions plus special functions according to the type of file and may be further enhanced using the Filetypes system.

Default Size: Allows you to set the initial size to use when opening a new Lister.

Output Window

This allows you to set the title and size, and device of the Output Window used by Opus 5 to display CLI tasks and associated messages.



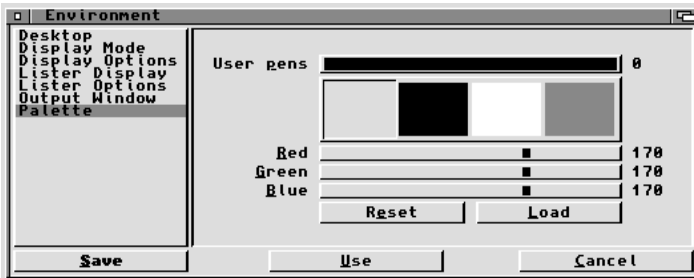
Title: This is the title name to be displayed in every output window.

Device: This is the name of the console device used for the output window. By default this is set to CON:, but you may refer to any other console devices which you may have installed, for example, KingCon or KCON:.

Size: To adjust the size, click on the *Set* button and adjust the position and size of the window as required, then close the window with the close window button.

Palette

The Palette displays the current colours used by the Opus 5 screen plus any user defined colours available.



On its own screen, the display will show up to 16 colours depending on the chosen screen depth. The first eight colours displayed are the Amiga OS colours, the bottom four and top four colours from your Workbench palette. Depending on the screen depth, you may also have up to eight user colours.

When on Workbench, only the user colours are displayed. The actual screen colours are determined from Workbench.

Reset: The reset button cancels any changes you may have made with the palette requester.

Load: The load button allows you to load any saved palette preferences file. Both Workbench 2.x and Workbench 3.x file formats are supported.

User Pens

The User Pens system provides a mechanism to make the management of colours practical and useful under V39 of the Amiga OS. (V37 is slightly different but Opus 5 provides similar functionality.)

With the pen allocation system of V39, it is possible to have your own colours even when running on another program's (or the Workbench) screen. The User Pens system allows you to define up to 8 colours which Opus 5 will attempt to allocate. You may use these on custom buttons and as the colour of Lister elements.

You can choose how many pens to allocate, from zero to eight. If you limit your choice to only those pens you actually need, all others are left for other programs. As well as the optional user colours, you always have the standard Amiga OS colours to use (4 colours under OS V37, 8 under V 39).

You can specify the desired number of user colours with the slider gadget. Once you have changed the number to what you want you need to select "Use". The Opus 5 window (and screen) will be closed and re-opened, and the new number of pens will be allocated (if possible).



Note that you can only change the colours of these pens if they were successfully allocated. If you slide the RGB sliders and nothing seems to happen, that's because there were no shareable pens available for Opus 5 to use.

Also note that you cannot have any User Colours unless your screen is at least 16 colours (8 under V37), since the Amiga OS grabs the top and bottom four colours for itself.

Problems with Screen Modes and User Pens

With Opus 5, some users became confused when changing screen modes after they had allocated user pens. If you have set up some user pens and subsequently change the screen mode to one of lower depth (less colours), where the same number of user pens cannot be allocated, Opus will still use the actual pen numbers you

had selected earlier for the display. This can cause some strange colour artefacts since these pen numbers will 'wrap-around' the number of colours available in your chosen screen mode. Once you have changed to a screen of lower depth, Opus may not let you change the number of user pens. This is not an error but is done by Opus 5 on purpose so as to preserve your previously defined colour selections for later use.

If, after you have allocated user pens, you wish to swap to a screen with fewer colours, set the User Pens back to zero *before* you change the screen mode. If you have set up a system of user pens and adjusted the colours exactly the way you want them, it is a good idea to save these Environment settings so you may easily recover your work at a later date.

This page is left intentionally blank.



Chapter Eight

The Options Editor

While the Environment controls the Opus 5 visual display elements, the Options settings provide control over many of the operational commands. These functions are:-

Caching

For speed of operation, Opus 5 can internally buffer each directory list. This is known as the **Directory Cache**.



Maximum number of dirs cached: Specify the number of internal buffers to be allocated for directories. You may use any value from one to 65535.

Disable directory caching: When selected, this turns off directory caching.

Re-read modified caches: When this option is selected, each cached directory is monitored. If the contents of a cached directory have changed when it is next activated, Opus 5 will re-read the directory.

Copy

This controls the actions taken when Opus 5 copies files.



Update destination free space: Once the copy has finished, this option causes Opus 5 to recalculate the free space remaining on the destination drive.

Set source archive bit: After Opus 5 copies a file, the archive bit of the original file will be set if this option is checked. This action can be used to indicate that the file had been archived.

Also copy source's: These flags tell Opus 5 that when it copies a file, it must also copy these parts of the file information as well. You may select all, none, or any combination of these options.

Delete

Major warning messages can be provided when deleting files and directories. Select as required from the following:-

Commencing delete: Tells Opus 5 to check with you before actually starting to delete. A very useful option. It's always better to be safe than sorry.



Deleting files: Tells Opus 5 to ask for confirmation before deleting every file.

Deleting directories: Tells Opus 5 to ask for confirmation before deleting directories.

Hide Method

These options allow you to specify and control the method Opus 5 will use to iconify itself.



Clock: Opus 5 will iconify to a one line clock display window on Workbench. To un-iconify the window, simply activate it by clicking on it with the left mouse button, then press the right mouse button.

Hotkey Only: Opus 5 will close all windows and not place any visual indication on the Workbench screen. To un-iconify, press the hotkey combination as defined in the **PopKey** field below.



The hotkey combination is always active, even when Opus 5 is not iconified.

Pressing the hotkeys will have the following effects:-

- if Directory Opus 5 is iconified, it will be un-iconified;
- if the Directory Opus 5 screen is open, but at the back of the display, it will be brought to the front;
- if the screen has been pulled down using the screen drag bar, it will be brought to the top;
- if the screen is already at the front of the display, it will be moved to the back.

AppIcon (Workbench Only): Opus 5 will open an icon on Workbench. Simply double click on the icon to un-iconify it. When iconified as an appicon, you can drop Workbench icons on the Directory Opus 5 appicon. This will have the same effect as if you displayed the file in a Lister window and double-clicked on it. For example, if you drop an IFF picture on the appicon, it will call the *Show* command.

AppMenu (Workbench Only): Opus 5 will add a menu to the Workbench's application *Tools* menu. To un-iconify, simply select the Directory Opus 5 menu item.

PopKey: Here you may enter the system hot key combination. If the string gadget is left blank, Opus will use the historic default combination of **Ctrl, left Shift, left Alt**.

With Opus 5.5, key sequences are automatically recognised. You may select any combination of Shift, Alt, Ctrl and Amiga qualifiers **plus a key**. If you wish to manually enter your own key sequence, depress the *Caps Lock* key first, then enter the desired command string. If you wish to delete a previously entered hot key, you will need to depress the *Caps Lock* key to enter string mode first, then clear the string. Note that if you choose other than the default combination, you cannot just enter qualifiers alone, you must enter a key as well.

Icons

These options define how Opus 5 processes files with associated icons or '.info' files.



Perform all actions on icons: When selected, whatever happens to a file will happen to that file's icon as well. For instance, if you delete the file DATA, and the file DATA.info also exists, it will be deleted too. If you rename the file FROG to BUFFALO, then FROG.info (if it exists) will be renamed as BUFFALO.info automatically.

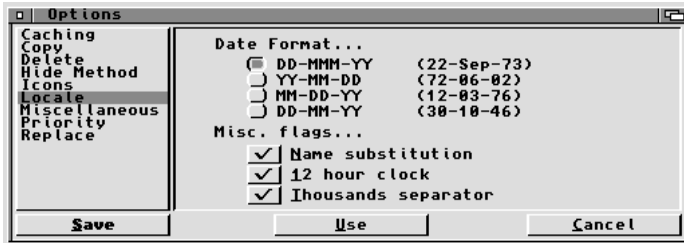
Select icons automatically: This option works in a similar way. Whenever you select a file, its associated .info file will also be selected (if it exists).

Trap 'More' in default tool: When you double-click or Open a project icon whose default tool is set to 'More', the file's contents will be displayed in the Opus 5 text viewer instead of invoking the More program.

Locale

Directory Opus 5 uses the Amiga Locale system to allow you to use Opus 5 in a variety of countries with different languages. This is determined automatically from the Locale options you have selected when you installed your Amiga OS and Workbench. Opus 5 also provides some extra control over the display of certain strings of data.

Options



Date Format: These options control the way Directory Opus 5 formats dates. The Amiga's operating system uses the first setting (DD-MMM-YY) by default, but you may choose whichever one you prefer.

Misc. flags...

Name substitution: Causes words like Today, Tomorrow or Tuesday to be substituted for a date, if appropriate.

12 hour clock: Chooses whether the clock format used by Directory Opus 5 is displayed in 12 hour or 24 hour format.

Thousands separator: Determines whether numbers have a separator between thousands. The character is determined from the current languages you have selected from the Workbench Locale installation.

Miscellaneous

This requester lets you set various special options.



Mouse buttons over inactive banks: With Opus 5.11, the window holding a Listers or Button Bank behaved in a manner similar to Workbench, where you have to activate the window before it can see any button events. *In Opus 5.5, this has changed!* With this option enabled, Button banks and Listers will see the right and middle mouse buttons even if their window is not active. Also, It is *now* possible to select a button from a button bank with the left mouse button without the button bank becoming active. This leaves the active focus on the current Lister.

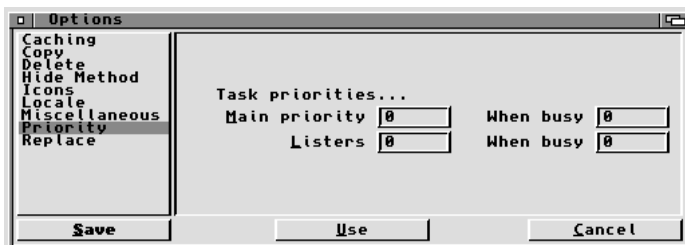
Quick quit: When set, Opus 5 will quit immediately when asked to without first asking for confirmation.

Extended lister key selection: With this turned on, you can quickly find files in large directories even if many files begin with the same letters. When you type the first letter, a string gadget will appear over the Lister toolbar enabling you to type as much of the filename as is necessary for it to scroll into view. As soon as you begin any other action the string gadget will disappear once more.

Filetype Sniffer: If enabled, Opus will automatically run the *FindFiletype* command when a file with no defined Filetype is double-clicked. With the option turned off, Opus will run the *SmartRead* command to view the file as either text, ansi or hexadecimal.

Priority

The priority requester allows you to customise the task priorities used by various parts of Opus 5.



Options

Main priority: This is the priority of the main Opus 5 process. You may specify a higher or lower priority for when Opus is busy.

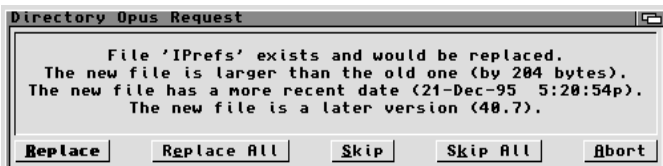
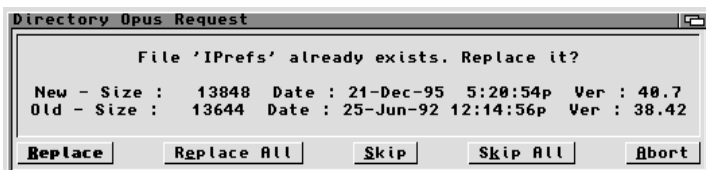
Listers: Each Opus 5 Lister runs a separate task which allows the smoothest multitasking possible. You may set the priority that all Listers will use for both normal and busy operation.



We recommend leaving the priorities at zero (0) unless you have a clear understanding of Amiga scheduling and a very good reason to change it!

Replace

This controls certain aspects of the requester which appears when copying a file over another with the same name.



Verbose 'Replace?' requester: Changes the requester from a simplified display to one with a more 'user-friendly' style showing extra information about the difference between the two files.

Automatically check version: This option invokes a version check when you attempt to replace an existing file and reports the results of the version comparison in the replace requester. If turned off, the version check may be performed manually from an extra button on the replace requester.



HINT: *When replacing large files it is recommended that you turn this option off, since searching for the version information on large files can take some time.*



The version check understands version information in libraries and devices and also looks for the version string as recommended in the Amiga Style Guide (This is defined as "\$VER: Name version.revision <(DD/MM/YY)>"). If the version information in the file does not follow the recommended standard, Opus may report anomalous results.

Obsolete Options - Options / Path Formats

In earlier versions of Opus, we used a special section called *Path Formats* where you could define a special format for a specific path plus add an special hot key for quick access to this path and format. With the extra display modes and other features of Opus 5.5, a separate section for this has become obsolete. However, you can still define specific path formats but in a more powerful way than before. Now, you can save not only the format for a Name Mode Lister but even the actual size, position and display mode for any desired path. This can be done as follows:-

Path Format for Name Mode Lister

Display the desired path in a Name Mode Lister, invoke the *Lister Format Editor*, set the required format, then select Save from the Format Editor.

Size, Position and Display Mode for any Lister

Display the desired path in a Lister in the required display mode, size and position the Lister to exactly where you wish it to appear on the screen, and take a direct snapshot from the *Lister/Snapshot* menu or from the Lister's RMB pop-up menu.

Hot Keys for a Special Path

In the *HotKey Editor*, add a new entry plus special hot key for your desired directory path similar to the following:-

Command SCANDIR WorkDir: MODE=name

or maybe

Command SCANDIR WorkDir: NEW MODE=icon

The *SCANDIR* command gives you the power to display your chosen directory exactly as required. Any special snapshots or previously saved display format will be used for this path.



Chapter Nine

Commands and Custom Buttons

With Directory Opus 5, you can create your own custom buttons which execute your desired commands at the click of the mouse. Buttons are grouped together in a window called a *Button Bank*. Any given bank of buttons may show either graphical images or text, but you may not mix graphics and text in the one bank. You may have as many banks open at any one time as you desire.

The modular design of Opus 5 means that Button Banks are fully independent tasks and the buttons themselves are interchangeable between different banks. Even editing of button banks via the *Button Editor* is independent of other program operations, so you do not need to stop other activities to create, load, edit or save banks.

Each button can execute an unlimited set of instructions, which may be triggered by the action of a left, middle or right mouse click on the custom button, or by selecting an item from the button's optional pop-up menu. Instructions may include any mix of AmigaDOS, Workbench, ARexx, Script or Opus 5 commands.

Right mouse button commands and pop-up menu commands are indicated by a '*dog-ear*' in the upper right of the button. For middle mouse button commands, the '*dog-ear*' appears on the bottom left. This also applies to graphical Button Banks.

With Opus 5.5, you can now define an infinite set of pop-up menus for each button. This allows you to group a larger number of

related functions in one place and select the specific one when required. Holding down the LMB on a button displays the full list of commands attached to that button. Releasing the LMB over a command brings this command to the top and makes it the default LMB command.

Scope and Focus of Buttons

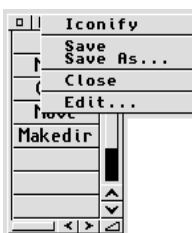
Many button commands act on the selected files in the current source (SRCE) directory, or between the source and destination (DEST) directories. When creating and using buttons, it is very important that you understand both what the command does, and what files and directories it will affect. This is known as the scope and focus of a button.

Novice Opus 5 users can be confused when they have clicked on a custom button and nothing happens. Often they have no SRCE window selected!



Before clicking on a custom button, verify that the intended SRCE and DEST directory windows are set correctly.

Special Buttons Menu



As well as from the global Buttons menu, Opus 5 also provides a special menu directly attached to each button window. This gives quick access to popular functions.

To access this menu, move the mouse to the button window title bar and press the right mouse button to display the menu. If you have '*Mouse buttons over inactive banks*' in *Options / Miscellaneous* disabled, you will need to activate the Button Bank window first.



The menu options are identical to those provided in the global Buttons menu, except that this special menu allows you to iconify the button bank to save space on your Opus backdrop window.

When iconified, the button bank will have a pop-up menu which will allow you to open the button bank again, or close it for good. You can also re-open it by double-clicking on the icon.

Internal Opus 5 Commands

As already explained, each button can execute a set of instructions including AmigaDOS, Workbench, ARexx, Script or internal Opus commands. The list of internal commands which Directory Opus 5 offers for each custom button or menu is shown below. Each command is discussed in detail in the following section.

Commands

AddIcon	FindFiletype	MoveAs
Alarm	FinishSection	None
All	Flash	Parent
AnsiRead	Format	Play
Assign	FreeCaches	Print
Beep	FTPAddressbook	PrintDir
CacheList	FTPCD	Protect
CheckFit	FTPCommand	Quit
CleanUp	FTPConnect	Read
CloseButtons	FTPCopy	Rename
CLI	FTPDelete	ReSelect
Comment	FTPQuit	Reveal
Confirm	FTPRename	Root
Configure	FTPSetVar	Run
Copy	GetSizes	ScanDir
CopyAs	HexRead	Search
CreateFiletype	Hide	Select
DateStamp	IconInfo	Set
Delete	Join	Show
DeviceList	LeaveOut	SmartRead
DiskCopy	LoadButtons	Split
DiskInfo	LoadEnvironment	Toggle
DoubleClick	LoadOptions	User1
DragNDrop	MakeDir	User2
Duplicate	MakeLink	User3
Encrypt	MakeLinkAs	User4
FindFile	Move	

- **AddIcon** **NAME, BORDER/S, NOBORDER/S, LABEL/S, NOLABEL/S, CHANGE/S**

Adds icons to all selected entries in the active Lister. Directory Opus 5 will automatically sense the type of file and the appropriate icon for a drawer, tool or project. The icons used are the current system default icons as defined in ENV:Sys directory. (See your AmigaDOS manual for more details.)

The **BORDER** and **NOBORDER** switches turn the icon border on or off. The **LABEL** and **NOLABEL** switches turn the (text) label on or off. If **CHANGE** is specified, the state of the borders or labels of existing icons will be changed without creating new icons.

AddIcon operates on all files selected in all current SRCE Listers.



Since this command uses the Amiga's default icons, it may fail if the correct default icons of the required type are not available in your ENV:SYS directory. If this happens, run the Amiga IconEdit program to create and save a set of default icons for the various file types. For more information on the creation and editing of default icons, please consult your AmigaDOS user manual.

- **Alarm**

This command compliments the *Beep* command and produces a short siren sound effect rather than a simple beep. Use this to provide a more noticeable or urgent warning sound than would be indicated by a simple beep.

See Also: Beep, Flash

- **All**

Selects all entries in any and all of the SRCE or selected Listers

See Also: None, Toggle, Reselect, Select

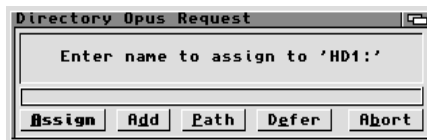
- **AnsiRead NAME/F**

Invokes the Opus viewer in ANSI mode which interprets and displays any special ANSI control sequences.

See Also: HexRead, Read, SmartRead

- **Assign**

Allows you to create AmigaDOS assignments for the path as shown in the Directory Path gadget at the bottom of the active SRCE Lister. A requester is opened with a set of buttons for :-



Assign: Create an assign for this directory with the specified name. (*Default*)

Add: Add this directory to the existing assign with the specified name.

Path: Add the path of this directory to the existing assign with the specified name.

Defer: Create a deferred assign for this directory which will become active only when first accessed.

Abort: Abort the assign operation.

For a more details discussion, refer to your AmigaDOS manual

- **Beep**

This command will sound a beep on either or both of the sound channels. You could use this command to signify that a command has finished.

For instance, you could create a *Diskcopy* button to read:-

Command DiskCopy
Command Beep

See Also: Alarm, Flash, FinishSection

- **CacheList** **NEW/F**

Displays a list of all the currently buffered directories. Click on one of the displayed buffers to jump to that buffer immediately, rather than having to locate it manually.

If there is no current SRCE Lister, a new Lister will be opened. If the **NEW** switch is used, a new Lister will always be opened.

See Also: DeviceList

- **CheckFit**

Tests whether the selected files will fit on the destination drive. A requester displays the number of bytes needed, the available space, and the percentage of the file which will fit on the destination drive.

CheckFit works between ALL SRCE directories to the first DEST directory ONLY.

- **CleanUp**

An Opus housekeeping command. It examines the internal 'position-info' file and remove references to any directories or left-out icons which no longer exist on your system. References will only be removed if the parent volume is mounted and available on your system so entries from removable media will be preserved.

- **CLI**

This is a new *debugging* command added for Opus 5.5 for the more advanced users. It opens up a Command Line Interface or Shell type window where you may enter internal Opus commands and ARexx functions directly. Such commands operate on highlighted files and between SRC and DEST Listers just as if you had them attached to a button or menu. The CLI is useful to test Opus commands or for quickly entering Opus commands which you have not set up in a button or menu. Type 'Help' in the window for more information.

- **CloseButtons** **NAME, ALL/S**

This command closes the current button bank, a named button bank, or all button banks. **CloseButtons** closes the bank from which the function was launched. **CloseButtons** <NAME> closes a named bank and **CloseButtons** **ALL** closes all button banks

See Also: LoadButtons

- **Comment** **NAME, COMMENT/F, RECURSE/S**

Adds or edits the comment field of all selected entries in the Lister. The maximum length of a comment is 79 characters. The **RECURSE** switch enables recursive access to any files in selected subdirectories, subject to the global *Recursive Filter*.

- **Configure**

Displays the *Lister Format Editor* for the current SRCE Lister.

- **Confirm** **TEXT/A**

Displays a simple requester showing your chosen **TEXT** and presenting the user with the choice of 'Ok' and 'Cancel' buttons. If the user selects the 'Cancel' button the Command script will be terminated at this point.

- **Copy** **NAME, TO, QUIET/S, UPDATE/S**

Copies the selected file(s) from the SRCE directory to the DEST directory.

If any directories are selected to be copied, the global setting of the *Recursive Filter* will determine which files are copied. See global menus *Settings/Recursive Filter* for more details.

If the **UPDATE** switch is specified, only files which do not already exist will be copied.

If multiple SRCE or multiple DEST directories are selected, the copy command will copy all selected items from the SRCE directories to all the DEST directories in turn.

- **CopyAs** **NAME, NEWNAME, TO, QUIET/S**

Copies the selected file(s) from the SRCE directory to the DEST directory using the new name that you specify. Wildcards can be used here; see the *Rename* command for more details. If any directories are selected to copy, the *Recursive Filter* determines which files will be copied.

Copies multiple SRCE to multiple DEST directories if more than one is selected.

- **CreateFiletype**

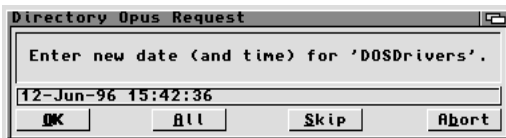
Invokes the *Filetype Creator* which can be very useful for automatically making Opus Filetypes for unknown files, especially if you do not know much about file formats. For detailed information on the *Filetype Creator*, see page 152.

See Also: FindFiletype

- **Datestamp** **NAME, RECURSE/S, DATE/F**

Allows you to change the datestamp of the selected files and directories in the active Lister. If directories are selected, you will be given to option to modify the datestamps of the files within them.

The **RECURSE** switch enables recursive access to any selected files in subdirectories, subject to the global *Recursive Filter*.

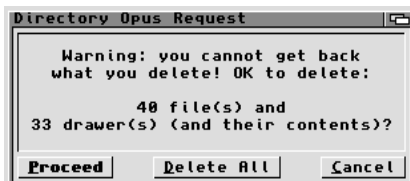


For each entry, you will be presented with a requester. To set the datestamp of the file to the current date and time, simply press return. Otherwise, enter the date and time you want.

Choose the 'OK' button, or press RETURN, to set the datestamp one file at a time. To set the datestamp of all selected entries at once, select the 'All' button from the requester.

- **Delete NAME, QUIET/S**

Deletes all the selected entries in ALL current SRCE Listers.



Be careful with this! It is easy to wipe out valuable data if you are careless. For safety, Directory Opus 5's default configuration displays a requester before deleting files. Nevertheless, you should always double check the selected files before clicking any button which uses this command.

The Delete command works on ALL SRCE Listers in turn.

- **DeviceList NEW/S, FULL/S, BRIEF/S**

Displays the list of all devices, volumes and assigned directories in the current SRCE Lister. If the **FULL** switch is

added, the list of Assigns is expanded to show the full path of the assigned volumes plus any multi-directory assigns. Alternatively, if the **BRIEF** switch is added, only the devices will be shown.

If there is no current SRCE Lister, a new Lister will be opened. A new Lister will always be opened if the **NEW** switch is specified.

When the device list is in Icon or Icon Action mode, you can delete and rename assigns using the icon menu.

See Also: CacheList

- **DiskCopy**

Invokes the Opus 5 *Diskcopy Requester* allowing you to select the source and destination drives and parameters for copying disks. (See page 185.)

- **DiskInfo**

Displays information on the parent disk of the current SRCE Lister. This command works only in a Lister, it does not work on a selected disk icon. For icons use the global menu item *Icons/Disk Information*.

This command works on the active SRCE Lister ONLY.

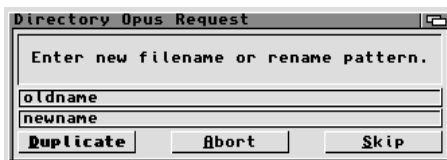
- **DoubleClick** NAME/F
- **DragNDrop** NAME/F

Execute the defined action for either DoubleClick or DragNDrop as defined in the Opus filetype for files of the selected type. (See *Filetypes* on page 139 for details.)

Commands work on all files selected in all current SRCE Listers.

- **Duplicate** NAME, NEWNAME, QUIET/S

Makes a copy of selected entries in the same directory, but with different names.



A requester will appear for each entry, asking for the new name. Wildcards can be used here; see the Rename command for more details.

The Duplicate command works on ALL SRCE Listers.

- **Encrypt NAME, TO, PASSWORD, QUIET/S**

Have you ever had files that you wanted to encrypt so that only people who knew the password could understand them? This command allows you to do just that. It will encrypt all selected files, using the password that you enter, with a complex algorithm that most people will find impossible to work out. The resulting files are not written over the originals, but are instead written to the destination directory. They will be the same size as the original files, so you can ensure you have enough room in the destination directory.



To decrypt a previously encrypted file, you should enter the same password and click on the *Decrypt files* checkbox, or precede the password with a minus sign. For example, to decrypt files encrypted with the password 'SMEG', select the files, choose the encrypt command and enter '-SMEG' as the password.

Encrypt operates on all files selected in all current SRCE Listers.

- **FindFile**

Searches all selected subdirectories in all SRCE Listers for a specified file or files. A requester will appear asking for the pattern to search for. You can use full AmigaDOS pattern matching for searches.

If a file matching the pattern is found, you may enter the directory containing it, or to continue the search. If you enter the directory, all matching entries will be highlighted.

- **FindFiletype**

This command invokes the *Filetype Finder* which can be used in finding, installing, and creating Filetypes for the selected files. For details, see the *FileType Creator* on page 152.



- **FinishSection**

Forces any preceding programs (AmigaDOS, Workbench, Batch or ARexx) to finish executing before carrying on to the next command. Note that the next command need not be an Opus command; it is just more likely that it will be one.

For instance, to add a beep to the end of the LHArc list Filetype command, you would change the command list to read:-

```
AmigaDOS LHArc v {f}  
Command FinishSection  
Command Beep
```

- **Flash**

This command has the same uses as the *Beep* command. It is different in that it calls the system *DisplayBeep* function so that it will flash the display and/or produce a beep or sampled sound as defined by the Sound preferences program.

See Also: Alarm, Beep

- **Format DRIVE/F**

Allows you to format a new disk. All new disks need to be formatted before the computer can write to them.

Without arguments, the command will display the Opus 5 *Format Requester* allowing you to choose the disk to format and other parameters. (See page 187.)

- **FreeCaches**

Clears all the internally cached directory buffers which are not currently displayed and frees all unused memory. If you are running a bit low on memory, this is a good way to free memory quickly.

If your system uses WB 3.0 or higher, Opus 5 will install its own internal low memory handler which will flush all unseen buffers automatically if required under low memory conditions.

- **GetSizes FORCE/S**

Causes any selected subdirectories in ALL SRCE Lister to be scanned to calculate the total size, in bytes, of all files contained in the subdirectory. Once scanned, the sizes of the subdirectory will be displayed in the Lister. It also displays in the status bar the number of files, directories and bytes that have been selected out of the total number of files, directories and bytes

If a subdirectory has been previously scanned, it will not be rescanned when you use the *GetSizes* command unless you use the **FORCE** switch.

GetSizes only checks the SRC Lister. If you wish to check if the selected files would fit on the DEST, use the *CheckFit* command.

If you select an operation which causes a subdirectory to be scanned (e.g., Copy, Protect, FindFile, etc.), the size will be displayed as though you had performed a *GetSizes*.

GetSizes works on ALL SRCE Listers in turn.

See Also: ClearSizes, CheckFit

- **HexRead** NAME/F

Reads and displays the selected files in the same way as *Read*, except in hexadecimal format. This allows you to view binary files and other files containing non-text characters.



Shown above is an example of the Hex Viewer's output. The first value is the offset, displayed in hex. This is the offset position, in bytes, from the start of the file. The next four values are each a four-byte long-word, with the actual ASCII representation at the end. Any non-text characters are shown with a period (.) character.

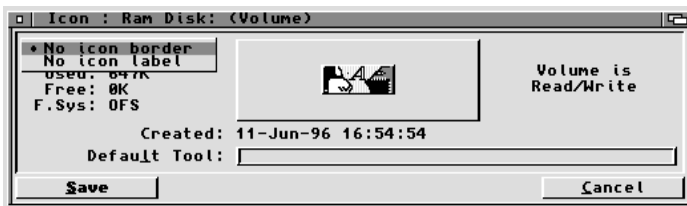
See Also: Read, AnsiRead, SmartRead.

- **Hide**

The hide command iconifies Opus 5. It has the same effect as selecting *Opus / Hide* from the global menus.

- **IconInfo NAME/F**

Allows you to modify the characteristics of icons such as stack size, default tool and Tool Types. It operates in a similar fashion to the Information menu of Workbench. With this command, you may select either the '.info' files themselves or the actual parent files or directories to which the icons refer.



A requester will appear when you run this command on a valid icon. The actual appearance of the requester will vary depending on the type of icon, but in all cases the actual icon imagery will be displayed. If you click on the icon imagery with the left mouse button, any alternative imagery will be displayed if it exists. The information displayed for each icon type is listed below.

Once you have made the desired changes to the icon, the *Save* button will save the changes to disk. The *Cancel* button will exit without modifying the icon on disk.

Drawer icon: For a drawer icon, you may edit the drawer's protection bits, comment and tool types. The date of the last modification of the drawer is also displayed.

Project icon: For a project icon, you may edit the project's stack size, default tool and tool types. Also displayed are the size of the project in bytes and blocks, and the last modification date.

Tool icon: For a tool icon, you may edit the tool's protection bits, stack size, comment and tool types. Also displayed are the size of the tool in bytes and blocks, and the last modification date.

Disk icon: For a disk icon, you may edit the disk's Default Tool. Displayed are the total number of blocks, and the number of blocks used and free. The block size, creation date, filesystem type, and read/write status are also displayed.

Trashcan icon: For a trashcan icon, you may edit the trashcan's protection bits, comment and tool types. The date of the last modification of the trashcan is also displayed.

Group icon: For a group icon, you may edit the group's comment and Tool Types. Also displayed is the last modification date.

Protection bits: (where appropriate) are modified in the same way as with the Protect command, except that the Hidden and Pure bits are not accessible.

Tool types: (where appropriate) are modified in the same way as from Workbench. To edit an existing Tool Type, simply select it, and press RETURN when you have modified it. To create a new Tool Type, select the *New* button. To delete an existing Tool Type, select it and then select the *Delete* button.

The IconInfo requester allows you to drop other icons on it to replace the imagery of the icon being edited. Clicking on any of the Tool Types toggles its state.

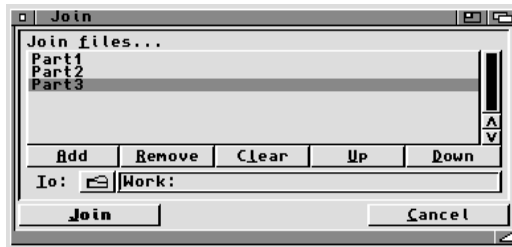
Icon Borders and Icon Name

New for Opus 5.5 is a special custom pop-up menu for the IconInfo requester, just press the RMB when over the IconInfo requester. The two options allow you to *turn off the icon border* and *turn off the display of the name* shown underneath the icon.

If you are running Workbench version OS3.0 or later and wish to use a third party replacement for the Workbench Information function, you can set the '*DOpus/UseWBInfo*' environment variable, and Opus 5 will call the OS WBInfo() routine for icon information (this may only work if you have Workbench running). Use this with patches like SwazInfo.

- **Join**

This command opens the Join Requester showing a list of the files that were selected in the SRCE directory. From the buttons, you may add and remove files to or from this list, clear the list, or rearrange the order of the files to be joined by moving individual files up and down in the list. Choose the destination directory, which defaults to the current DEST directory if one is present, and press the *Join* button. You will then be prompted for a filename for the new file.



See Also: Split

- **LeaveOut NAME/F**

Places the currently selected files on the Opus 5 Main Window and permanently leaves them out for easy access.

The command works on all selected entries (files or directories) in all current SRCE Lists.

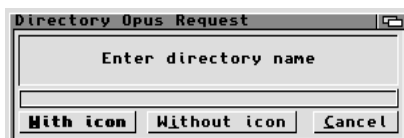
- **LoadButtons** **NAME/F**
- **LoadEnvironment** **NAME/F**
- **LoadOptions** **NAME/F**

These three commands take a filename as an argument and load the Opus 5 component files as described.

LoadEnvironment and *LoadOptions* will then reset the program operation to the newly loaded parameters. If only a simple filename is given, each command searches in the appropriate Opus 5 path of either *DOpus5:Buttons/* or *DOpus5:Environment/* or *DOpus5:Settings/* for the specified file. If a full pathname is given, the command will use that pathname instead.

- **Makedir** **NAME, NOICON/S, SELECT/S, NEW/S, READ/S**

Allows you to create a new subdirectory in the SRCE directory window; its name is limited to 30 characters.



From the requester, you can choose whether to create an icon along with the new directory or not. Entering a name and just pressing RETURN, will either create an icon, or not, according to the status of the global *Settings/Create Icons?* menu. The name of the directory is limited to 25 characters if Create Icons is enabled. The newly created directory will always be scrolled into view for further use when in Name mode.

If the optional **NAME** is used, the command will not ask for the directory name, but will make it immediately in the current SRCE directory.

If the optional **NOICON** switch is used, the directory will be made without an associated '.info' file.

The **SELECT** switch will cause the newly created directory to be selected immediately, and the **READ** switch will cause it to be read into a new Lister.

MakeDir works on the first SRCE or current Lister ONLY.

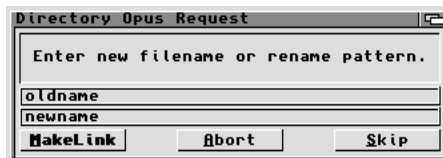
- **MakeLink NAME, TO, QUIET/S**

This command creates a HARD link to files and directories from SRC to DEST. In essence, it is similar to the *Copy* command but creates an Amiga link file in the DEST instead of copying the file itself. (See the AmigaDos manual for more information on links.)

By a limitation of the Amiga filesystem, hard links are only supported to files and directories on the same volume. Soft links are not supported by Opus 5 and cannot be created with this command.

To distinguish them from plain files and directories, links are displayed in bold in Listers in name mode. In icon mode, they have a little 'arrow' image superimposed on them to indicate that they are links. The same image is shown on any icons which have been left-out.

Makelinks works from multiple SRCE to multiple DEST directories if more than one is selected.



- **MakeLinkAs NAME, NEWNAME, TO, QUIET/S**

Performs the same command as *MakeLink* but allows you to give each entry a new name in the destination directory. Wildcards can be used here; see the *Rename* command for more details.

MakelinkAs works from multiple SRCE to multiple DEST directories if more than one is selected.

- **Move** **NAME, TO, QUIET/S**

Moves all selected entries from the SRCE directory to the DEST directory. The entry will no longer exist in its original place. If any directories are selected to move, the *Recursive Filter* will be used to determine which files will be copied.

If the *Move* operation is on the same device, Opus 5 actually uses the *Rename* command. On different devices, *Copy* and *Delete* commands are used.

Move acts to move files to a single destination ONLY.

See Also: MoveAs, Copy, CopyAs



Be careful with this command! Opus 5 will delete the file if you are moving it to a different device.

- **MoveAs** **NAME, NEWNAME, TO, QUIET/S**

Performs the same command as *Move*, but allows you to give each entry a new name before it is moved. Wildcards can be used here; see the *Rename* command for more details.

MoveAs acts to move files to single destination ONLY.

See Also: Move, Copy, CopyAs

- **None**

Deselects all entries in all the SRCE directory windows.

See Also: All, Toggle, Select, Reselect

- **Parent**

Reads and displays the parent directory of the current directory open in the SRCE Lister. If the parent directory is contained in Opus 5's cached buffer list, it will be displayed without re-

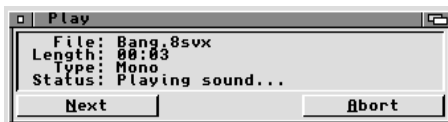
reading it. The command is assigned to '/' and '?' keys in Listers. The shifted '?' key opens a new Lister for the parent.

When the current directory is an assignment, for example, C:, the assignment will be expanded to the full path (Workbench:C), whereby the parent command will move to the parent directory (Workbench:) of the assigned directory.

This command acts on the first SRCE directory only.

- **Play NAME, QUIET/S**

Allows you to listen to sound files. This command plays IFF 8SVX format sound files, raw data files, and a limited set of SoundTracker type MOD files. It will also play other sound formats via the datatypes system of OS3.0 and higher. Unless you have specified the **QUIET** option, A small requester will appear while the sound is playing, showing the name, type of sound file, and playing time. To abort a sound before it has finished playing, click the 'Abort' button in this requester, or click the 'Next' button to skip to the next sound.



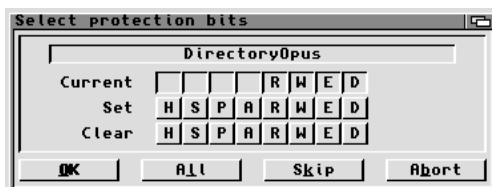
Because of some deficiencies in the OS datatypes system, Opus sometimes cannot tell when a sound being played through datatypes finishes playing. If this is the case, you will have to click the 'Next' or 'Abort' button manually.



Due to the explosion in the variety of sound module formats, e.g. Star/Sound/Noise/ProTracker, Med, OctaMed, Octalizer, and Med with MIDI modules, Opus 5 cannot play all such sound formats. To play these formats, we recommended that you set up a button to call one of the many excellent sound players currently available. (DeliTracker is an excellent example and can be readily obtained from AmiNet.)

- **Protect** **NAME, RECURSE/S, SET=+/K, CLEAR=-/K**

Modify the protection bits of the selected files and directories in the active directory window. When you select directories, you will be asked whether you also wish the files within them to be protected. For each entry, you are presented with a requester displaying the protection bits currently set for that entry. This is described in greater detail later.



The **RECURSE** switch enables recursive access to any selected files in subdirectories, subject to the global Recursive Filter .

Protect applies to all selected files in all the current SRCE Listers.

The Protection Requester allows you to change the protection bits of a file or subdirectory. The protection bits are a group of flags stored with the file, that determine the characteristics of the file. These flags are given single character names. The protection bits currently in use are HSPARWED.

H Hidden: If this flag is set, the file is not normally displayed. This allows you to mark certain files as "invisible", to avoid cluttering your directories. The file can still be accessed normally, and not all programs implement this flag.

S Script: A script file is a file containing a list of AmigaDOS commands to execute; it is like a simple computer program. This flag indicates that the file in question is a script file. A script file is sometimes called a batch file.

P Pure: If a program file is flagged as pure, it can be made to remain in memory ("made resident"), even when not in

use. This can save a great deal of time, especially if the program is used often, since it does not have to be loaded from disk each time.

A Archive: This flag indicates that the file has not been changed. If this file is ever written to, the 'A' flag will be cleared. This flag is often used by hard disk backup programs to record which files have not been changed and do not need to be backed up again.

R Readable: If this flag is set, the file can be accessed.

W Writeable: If this flag is set, the file can be written to (ie, more information can be stored in it than is already there).

E Executable: If a program file does not have this flag set, it can not be run.

D Deleteable: If this flag is not set, the file can not be deleted.

The Protection Requester shows the current file to be modified and the protection bits currently set. Underneath are two rows of buttons corresponding to the protection bits which you may wish to set or clear. When a button is highlighted, it means that the bit will be cleared or set as shown when you click the 'OK' button.

The bottom of the requester gives you buttons to chose the action required for this specific file.

OK: Causes the current file's protection bits to be set as indicated in the display.

All: Causes all selected files to be set, without additional prompting, as indicated in the display.

Skip: Skips over the current file and moves on to the next selected file in sequence.

Abort: Aborts the *Protect* command.

- **Print** **NAME/F**

Prints the selected files from all the current SRCE directories. It first displays the Opus 5 *Print Requester* which allows full control over print formatting.

New for Opus 5.5, the Print routine automatically prints a form feed after every file. See page 189 for more details of the *Print Requester*.

- **PrintDir**

Print the current directory list shown in the SRCE Lister. The directory will be printed as currently displayed. To change the format of the print-out you must edit the Lister format first. PrintDir works via the *Print Requester*, giving control over print formatting.

PrintDir operates on the current SRCE Lister ONLY.

- **Quit** **FORCE/S**

The Quit command will quit Opus 5, first asking for confirmation unless you have set *Quick Quit* in the *Options / Miscellaneous* settings. With the **FORCE** switch, Opus will always quit without asking for confirmation.

- **Read** **NAME/F**

Displays the Opus 5 text reader so you may read selected files. The name of the file is displayed in the viewer window's title bar. Since the viewer is an independent program with its own window, you may open as many viewers to show as many different files as you wish at any one time.

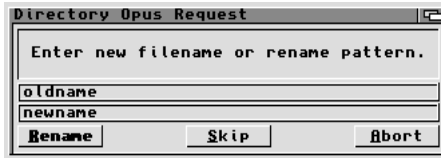
The Opus 5 viewer provides a number of options from its menu selections. These include search and print capabilities and are discussed in detail on page 192.

Read operates on the current SRCE directory ONLY.

See Also: AnsiRead, HexRead, SmartRead

- **Rename**

Allows you to give new names to all selected entries in the currently selected SRCE directory windows. A requester will appear for each entry in turn, asking for the new name. The initial rename requester has two string fields instead of one. You will usually just edit the name in the lower of the two to the new name.



A limited type of wildcard rename is possible. Entering an '*' in the bottom field allows you to add prefixes or suffixes. For instance, entering '*.pic' will add a '.pic' suffix to all selected entries. Entering 'A*' will add an 'A' prefix. Only one '*' may be used in this process.

You cannot give a file a name that contains an ''.*

If you enter an '*' in the top as well as the bottom field, you can replace sections of the name. for example, entering '*.pic' in the top field and '*.iff' in the bottom field will replace the '.pic' suffix of any entries with the '.iff' suffix. If an entry does not have a '.pic' suffix, it will be left untouched. The '*' may also be embedded.

For instance, renaming 'FOO*BAZ' as 'GEE*WIZ' would rename 'FOOBARBAZ' to 'GEEBARWIZ'. Again, only one '*' may be used in each of the string Fields.

The Rename command works on ALL SRCE Listers in turn.

- **Reselect**

The *Reselect* command causes all entries in all SRCE directories which were deselected by the operation of the previous command to be reselected.

The *Reselect* command is assigned to the ``\` key in Listers.

See Also: Select, All, None, Toggle

- **Reveal**

This command causes Opus 5 to deiconify if it has been iconified. It will also bring the Opus 5 screen to the front or send it to the back if it is already the frontmost screen. It is equivalent to pressing the hot key combination (by default, Ctrl + left Shift + left Alt).

- **Root**

Reads and displays the root directory of the SRCE Lister. As with the *Parent* command, the buffer list will be searched before the parent directory is re-read.

When the current directory is an assigned directory, the *Root* command will display the root drive of the assigned directory. When the current directory is a subdirectory to an assigned directory, the *Root* command will display the assigned directory. It is assigned to the `:` and `;` keys in Listers.

This command acts on the first SRCE directory only.

See Also: Parent

- **Run** **NAME/F**

Runs each selected file in turn, providing that the file is an executable program. The action is similar to double-clicking on the file's icon, or running it from the CLI. A requester will appear, asking for any arguments (should you require any).



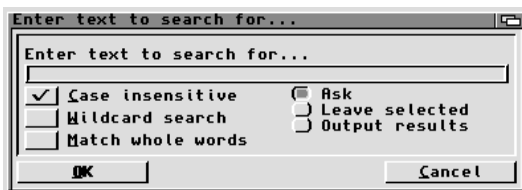
Directory Opus 5 does not provide an 'Execute' command. This function is handled by the Run command which functions in the same manner as the AmigaDOS 'run' command. If a file has the 's' (script) bit set, the run command will execute it as a script.

- **ScanDir** **PATH,NEW/S, MODE/K, SHOWALL/S**

With no arguments, re-reads the current directory in the first SRCE Lister. If you specify a path, that path will be read into the current SRCE Lister. If there is no current SRCE, it will open a new Lister. If the **NEW** switch is used, it will always open a new Lister. You may specify the display mode with **MODE=ICON, ACTION** or **NAME**, otherwise the new Lister will open using the system default as defined by the settings of the global *Lister/ViewAs* menu, unless overridden by a stored snapshot of this path. The **SHOWALL** switch will cause files without icons to be visible when in an icon mode.

- **Search**

Searches the contents of all selected files in all SRCE Listers, and the files within selected directories, for a specified string. A requester will appear, asking for the search string. There are also several options available within the requester.



Case insensitive: When checked, any upper and lower case letters are treated as the same.

Wildcard search: When checked, you can use a question mark (?) character as a limited wildcard, to mean "match any character".

Match whole words: When checked, the string you enter must be matched in a whole word (if the string is embedded within a longer string it will not be matched).

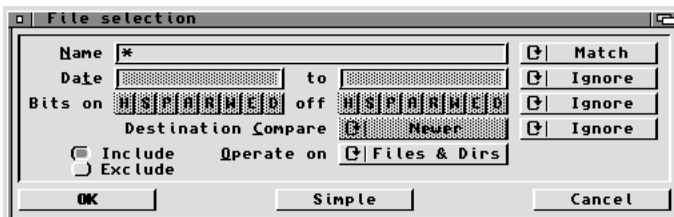
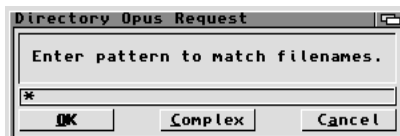
You can also specify how the results of the search are presented to you.

Ask: If a match is found, Opus 5 will ask whether you wish to read the file. If selected to do so, the text viewer will be opened and the file will be read automatically.

Leave selected: If enabled, any files that contain the matching string will be left highlighted. All files that did not match will be deselected, enabling you to see quickly the files that contained a match.

Output results: If enabled, the names of files that contained the matching string will be written to a temporary file. Once the search has finished, this temporary file will be displayed in the text viewer.

- **Select** NAME/K, FROM/K, TO/K, BITSON/K, BITSOFF/K, COMPARE/K, MATCHNAME/S, NOMATCHNAME/S, IGNORENAME/S, MATCHDATE/S, NOMATCHDATE/S, IGNOREDATE/S, MATCHBITS/S, NOMATCHBITS/S, IGNOREBITS/S, MATCHCOMPARE/S, NOMATCHCOMPARE/S, IGNORECOMPARE/S, BOTH/S, FILESONLY/S, DIRSONLY/S, EXCLUDE/S, INCLUDE/S



When called with no arguments, *Select* displays a requester allowing you to specify a pattern to match files in the current SRCE Lister. Files matching the selection criteria will be selected or deselected depending on the state of the Include or Exclude switch.

The optional arguments take their names from the fields displayed in the complex selection requester. If called with arguments which satisfy a selection criteria, the requester will not be displayed.

The selection requester may be used in simple or complex mode as shown above.

See Also: All, None, Toggle, Reselect

- **Set**

The *Set* command allows you to set certain things regarding the current Lister or the operation of the current function.

Set Output <handle> - allows you to define the output handle for the lifetime of this function where <handle> is CON: or a filename etc.

Set Sort <method> - changes the sort method for the current source Lister where <method> is either NAME, SIZE, DATE, PROTECTION, COMMENT, FILETYPE, VERSION, OWNER, GROUP, or NETPROT (for access).

Set Display <items> - changes the display items (same options as Set Sort above).

Set Separate <method> - changes the file/directory separation where <method> is either MIX, DIRSFIRST or FILESFIRST.

Set Hide <pattern> - changes the hide pattern.

Set Show <pattern> - changes the show pattern.

Commands and Custom Buttons

Set Flags <flags> - sets various flags of REVERSE, NOICONS, or HIDDEN.

The '*Set Flags*' command allows you to add and subtract flags. For example, "*Set Flags +hidden -noicons*" sets the hidden flag, clears the noicons flag, and leaves all other flags untouched. The state of a given flag can be toggled by using the '/flag' command, e.g. *Set Flags /hidden*

Set ToolBar <name> - allows you to change a Lister's toolbar.

Set Mode <mode> - sets mode to NAME, ICON, or ACTION, with or without SHOWALL switch.

Set Source - sets Lister into source mode.

Set Dest - sets Lister into destination mode.

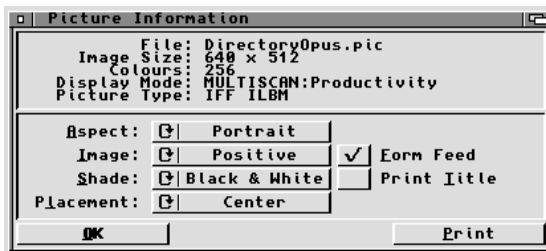
Set lock [on | off] - locks or unlocks Lister as SRCE/DEST.

Set off - turns Lister off.

As with other Opus commands, the *Set* command and options are not case sensitive.

- **Show NAME/F**

Displays IFF ILBM pictures, brushes and animations. It will also display other picture formats via the datatypes system of OS3.0 and higher.



Opus 5 will show most pictures and brushes, including overscan, extra halfbrite (EHB), HAM (4096 colour) pictures, and AGA 8 bit pictures.

Under OS3.0 and higher, if a file is not in IFF format, but in a format for which a datatype has been installed, the picture will be displayed by that datatype.

You may scroll around the image by moving the mouse pointer and the following actions or keys can be used when viewing a picture or animation:-

Space, Help or **P** for Help and Print Requester,
Esc or **RMB** to abort,
LMB to deselect this picture and view the next,
Q to view the next image without deselecting this one,
Del to view the next image and mark this one for deletion.

These keys can be used when viewing an Animation:-

S	Starts and Stops
N	Next Frame
-	Slow Down
=	Speed Up
\	Original Speed
F1-F10	Various Speeds (F1=Fastest)

If any pictures are marked for deletion when the last picture has been shown, the *Delete* function will be automatically launched on them (you are given the usual warning first).

The *Show* command will use its own IFF code in preference to datatypes IFF, since datatypes can be slower and have more problems. If you wish to force Opus to use datatypes first, set the Opus environment variable with:-

setenv dopus/ShowUseDatatypesFirst

- **SmartRead NAME/F**

Invokes the Opus 5 reader program in either text, ANSI or HEX mode according to the type of file selected. As with the read command, if multiple files are selected, they will be displayed in sequence. Pressing ESC will terminate the reading of the sequence. See page 192 for more information regarding the reader.

See Also: AnsiRead, Read, HexRead

- **Split**

Splits files into smaller chunks. It opens a requester for you to pick the file to split and the destination directory in which to put the parts. You may choose the size of the chunks into which to split the file from preset values optimised for Double Density floppy disk size, High Density disk size, Double Density MS-DOS disk size, High Density MS-DOS disk size, or a custom size of your choice. Size is measured in kilobytes. You must also provide a stem name for the output files to which .000, .0001, etc will be added.



If the destination directory is a floppy disk (either a trackdisk.device or a mfm.device disk), you will be prompted automatically for a new disk to be inserted as necessary.

Split works on ALL SRCE Lists in turn.

- **Toggle**

Causes selected entries in all SRCE directories to be deselected and deselected entries to be selected. This command is often used as a complementary right mouse button action for a button using the *All* command. The right mouse button would then toggle or reverse the state of all entries.

See Also: All, None, Select, Reselect.

- **User1, User2, User 3, User 4 NAME/F**

These commands invoke the four user definable commands associated with your user defined Opus Filetypes.

For example, with the LHA Filetype supplied with Opus 5, the *User1* command is defined to extract files from the archive. So, selecting an LHA file and pressing a button with the command, *User1*, would reference the Filetype, obtain the command and extract the archive.

This page is intentionally left blank.



Chapter Ten

The HotKey Editor

The Hotkey Editor allows you to attach functions to any keystroke combination. Hotkeys are effectively the same as standard Opus buttons which are called directly by the press of a key rather than with a mouse click. Although you could always attach hot keys to the traditional Opus buttons and menus, Opus 5.5 now allows you to define a specific set of independent hot keys. As with Opus buttons, menus and scripts, a Hotkey may define a set of instructions which includes any mix of AmigaDOS, Workbench, ARexx, Script or internal Opus 5 commands.

Hotkeys can be **Local**, only accessible from within Opus 5, or **Global**, accessible from anywhere in the system as long as Opus 5 is running.



Items: Shows the list of currently defined Hotkeys. On the left is the name of the Hotkey function. On the right is actual hot key combination that triggers this function.

Item Name: This allows you to enter or modify the name of the Hotkey you have selected from the list above.

Add: Adds a blank Hotkey definition at the end of the list.

Insert: Adds a blank Hotkey definition before the one currently highlighted in the displayed list.

Duplicate: Adds a copy of the currently selected Hotkey definition at the end of the list.

Delete: Remove the selected Hotkey definition.

Edit: Calls the *Function Editor* which allows you to make changes to the selected Hotkey. You can also edit a Hotkey definition by double-clicking on its entry.

System-global Hotkey: Installs the selected hotkey into the Amiga Commodities system which makes it available throughout the system. By default, this is not selected and the hot key will be active only when Opus is the active program.

Save: Saves the currently displayed set of Hotkeys to disk and updates the Hotkeys used by the system. Note that this saves the Hotkeys using the 'current' filename, i.e. the same name under which they were last loaded. Any previous file of this name will be overwritten.

Use: Updates the Hotkeys used by the system but does not save them to disk.

Cancel: Cancels all changes you have made to the Hotkeys.

The Hotkey Editor Menus

As with the other editors in Opus 5, there are extra options provided by menus. These are:-

The Project Menu

New: Creates a new blank Hotkey list.

Open: Displays a file requester allowing you to load a new set of Hotkeys.

When first run, Opus uses a default filename for the Hotkeys. Once you load a new set of Hotkeys using a different name, this new name will be kept and used internally as the reference to that set of Hotkeys. If you subsequently save the particular set of Hotkeys, it will be saved under this name unless you use the SaveAS option. If you save the Environment, this 'new' filename will be stored with the environment.

Save: Saves the displayed set of Hotkeys to disk under the current name.

SaveAs: Saves the current set of Hotkeys but allows you to select a new filename.

Quit: Same as Cancel above.

The Edit Menu

Reset to Default: Resets the Hotkeys to the default set as defined when you installed Directory Opus 5. Because there are many settings, these defaults are not actually built-in to Opus 5, instead Opus will look for and load special default files.

Last Saved: Reloads the last saved set of Hotkeys and resets the display.

Restore: Restores the Hotkeys to the state when you first opened the Hotkey Editor.

Sample HotKeys

A simple example of a Hotkey would be to display a specific directory or disk. For example, add a Hotkey as follows:-

Name:	Read "RAM"
Key:	lalt r
Command:	SCANDIR "RAM:"

Pressing the LALT plus the 'r' key will display the contents of your Ram Disk.



Chapter Eleven

The Scripts Editor

Scripts are a new feature for Opus 5.5. A Script is a special type of Opus button which is invoked when a specific event occurs. As with Opus buttons, menus and hotkeys, Scripts may define a set of instructions which include any mix of AmigaDOS, Workbench, ARexx, script or internal Opus 5 commands.

Opus can call a Script whenever any of the following system events occur:-

Bad disk inserted: Activated whenever a disk of unknown format is inserted into any floppy drive. As an example, this Script could be used to prompt the user to format the disk.

Close buttons: Activated whenever a Button Bank is closed.

Close group: Activated whenever a Group window is closed.

Close lister: Activated whenever any Lister is closed.

Disk inserted: Activated whenever a disk of known format is inserted into any floppy drive. As an example, this Script could be used to play a sound file or make a beep when you insert a disk. Alternatively it could automatically open a lister and display the contents of the inserted disk. Or even both?

Disk removed: Activated when a disk is removed from any floppy drive. Maybe use this to play a different sound?

Double-click: Activated when the LMB is double-clicked on any clear part of the Opus 5 Main Window, but it is not activated when the mouse is over any other Opus element such as a lister or button bank etc. One handy use for this Script is to open a new lister or device list as shown below.

Hide: Activated when Opus 5 is iconified.

Middle double-click: Activated whenever the MMB is double-clicked providing the mouse is not over any button which has a MMB function defined. This Script will even be activated if Directory Opus is not the active process.

Open buttons: Activated whenever any Button Bank is opened.

Open group: Activated whenever any Group window is opened.

Open lister: Activated whenever a Lister is opened.

Reveal: Activated when Opus is de-iconified, by pressing the hot key combination, Ctrl + lshift + lAlt, for example.

Right double-click: Activated whenever the RMB is double-clicked and the mouse is not in a position which would normally activate a menu or a RMB function for any button.

Shutdown: Is executed immediately before Opus 5 quits.

Startup: Is executed when Opus 5 is first started.

The Editor

The Script Editor displays the names of all of the events supported by Opus and allows you to add, edit or delete the command instructions attached to each event. An event which has attached commands is shown in white (Colour 1), while events with no defined commands is shown in black (Colour 2).



Scripts: Displays names of the available Scripts. Scripts which have functions attached are shown in a highlighted colour.

Delete: Deletes the function associated with the selected Script.

Edit: Allows you to edit the functions attached to the selected Script with the *Opus Function Editor*. The Function Editor can also be opened by double-clicking on the desired Script.

Save: Saves the currently displayed Scripts to disk and updates those currently used by Opus. Note that this saves the Scripts using the 'current' filename, that is, the name under which the Scripts were loaded. Any previous file of this name will be overwritten.

Use: Updates the Scripts currently used by Opus but does not save them to disk.

Cancel: Cancels all changes you have made to the Scripts.

The Script Editor Menus

As with the other editors in Opus 5, there are extra options provided by menus. These are:-

The Project Menu

New: Creates a new blank Script list.

Open: Displays a file requester allowing you to load a new set of Scripts.

When first run, Opus uses a default filename for the Scripts file. Once you load a new set of Scripts using a different name, this new name will be kept and used internally as the reference to those Scripts. If you subsequently save the particular set of Scripts, they will be saved under this name unless you use the *SaveAS* option. If you save the Environment, this 'new' filename will be stored with the environment.

Save: Saves the displayed Scripts to disk under the current name.

SaveAs: Saves the current Scripts but allows you to select a new filename.

Quit: Same as Cancel above.

The Edit Menu

Reset to Defaults: Resets the Scripts to the default set as defined when you installed Directory Opus 5. Because there are many settings, these defaults are not actually built-in to Opus 5, instead Opus will look for and load a special default file.

Last Saved: Reloads and installs the last saved set of Scripts.

Restore: Abandons any changes you have made and restores the Scripts to the state as when you first opened the Script Editor.

Sample Scripts

Opus 5.5 comes with just two simple Scripts installed. These are :-

Double-Click: This is set to call the command '*DeviceList NEW*' which will open a new Lister whenever you double-click the LMB on the Opus Main Window.

Middle Double-Click: Runs the AmigaDos command NewShell to open a new Shell window on either your Opus screen, if it has been set to *Default Public Screen*, or the Workbench screen.

There are many easy and simple uses for Scripts. Try the following for the disk inserted and disk removed Scripts. Add the command, '***PLAY QUIET <sound>***' where *<sound>* is the full file path to a favourite sound effect.

This page is left intentionally blank.



Chapter Twelve

Filetypes

A file is simply stored data. Files can contain executable programs, text files, pictures or various types, icons for Workbench, or a multitude of other kinds of data. Most, but not all files, have an identifiable structure. Directory Opus 5's Filetypes system is designed to examine a file's internal structure and identify the type of data it contains. You can configure Directory Opus 5 to understand an unlimited number of Filetypes.

Filetypes are a powerful feature of Directory Opus 5. By using Filetypes, you can configure Opus 5 to play animations when they are double-clicked, to load and use Multiview when you double-click on an AmigaGuide help file, or to uncompress an archived file when you drag and drop it to a new directory or even to feed the archive to an external handler such as *ArcDir*.

This is the essence of the Filetypes; when you do something to a file, Opus 5 can figure out what kind of file it is, and take the appropriate action for that type of data.

Predefined Filetypes

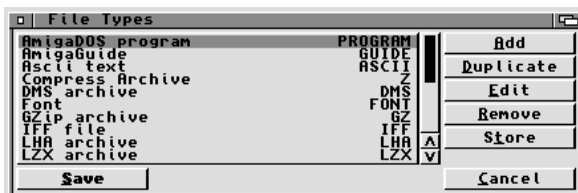
Opus 5 comes with some fully defined Filetypes. These are stored in the *DOpus5:Filetypes* directory and have both the Filetype structure fully defined, plus have default commands attached to the various actions available. The loading of Filetypes is dynamic. Opus 5 will look to see what Filetypes you have defined in this directory and load them automatically.

Filetypes

For convenience, we have also provided a set of default Filetypes definitions in the *DOpus5:Storage/filetypes* directory. With these, we have done most of the hard work and have set out the details needed by Opus 5 to recognise files of the specific types. To use one of these pre-defined Filetypes, you should use the *Automatic Filetype Creator* to identify and install the matching Filetypes or you can manually drag the required one into the *DOpus5:Filetypes* drawer and Opus 5 will recognise and load it automatically. Then, edit the new Filetype and set out the specific actions you wish to attach to the new definition. By doing this, you don't have to be an expert or know anything about the internal structure of the various files. All you need to do is attach the actions which Opus 5 should take when it recognises a file of this type.

Filetype Manager

The Filetypes requester is accessed from the Settings menu and displays the list of Filetypes that Directory Opus 5 recognises. For starters, we have included several definitions. These may include AmigaGuide, LHA archive, Workbench icons, and Script files. Your currently defined and available Filetypes will be shown in the list in alphabetical order. For example,



Add: Allows you to create a new Filetype entry based on a predefined File Class. When you select this button, two requesters appear where you may define the actions and definitions of the new Filetype.

Duplicate: Allows you to quickly duplicate a current entry. Highlight the desired entry, select *'Duplicate'* and a new entry will be cloned from the current one and presented for editing.

Edit: Displays the Filetype events and allows you to edit the Filetype definition and command actions.

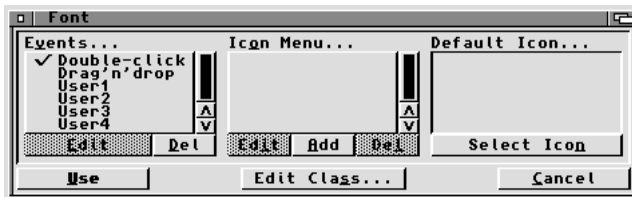
Remove: Removes a Filetype definition from the list and deletes the entry from the *DOPus5:Filetypes* drawer.

Store: Removes a Filetype definition from the list and moves the entry from the *DOPus5:Filetypes* drawer to the *DOPus5:Storage/Filetypes* draw.

Editing Filetypes

The Filetype Editor consists of a number of parts; one showing the actual Filetype definition or class, one showing the possible events or user actions, and one detailing the corresponding commands each action will perform.

Double-clicking on a particular Filetype, or selecting the *Edit* button, displays the list of events or user actions which can be defined for each Filetype, a special set of menus which you may attach to the sticky menus for icons of this type, and the default icon image you wish to use for this type of file.



Events

Each of the actions is associated with either a mouse event or a limited set of Opus 5 commands, namely User1 - User4.

A tick on the left of an entry indicates that it already has an Function Command List defined for it. Clicking on an entry in the list displays the *Function Editor*, allowing you to edit the commands associated with this event.

Filetypes



To compare the definitions for multiple events, or edit multiple definitions simultaneously, double-click on multiple events in turn. While each of these event types can be defined to do something different, usually only a few are actually defined. It is certainly not necessary to define all events for a particular Filetype.

When one of these actions or events occurs, Opus 5 does the following:-

- It first searches the Filetypes list, starting with the Filetype of highest priority, and checks whether it matches the entry's Filetype definition.
- If it matches, it checks if the corresponding event has been defined as a notifiable action. If it is defined, it performs the attached function.
- If there is no match for the Filetype definition, or if there is a match but no associated function, it continues to search the Filetypes list. Note that it is possible for it to match a subsequent entry that has a function defined.

Mouse Events

Mouse events occur when you either double-click on a file or when you drag and drop it into a new directory.

Double-click: This occurs when you double click on a file. A popular use of this action is to examine a file and, for example, to show it if it is a picture, or play it if it is a sound. The actual double-click speed is defined by your Amiga OS preferences.

Drag and drop: This event occurs when a file is clicked on, dragged to another window and released. One popular use of this event is for extracting an archive.

Command Events - User 1-4

Command events are called when a file is acted upon by a limited set of Directory Opus 5 commands. Only the User event commands are available. The terms User 1, User 2, User 3, and User 4 may seem cryptic, but they are here to give you flexibility. Each of the normal Opus 5 commands has an implied usage, but you may have an application which doesn't really mean any of these. In that case, you can decide that one of these User events means "Perform this special operation".

Icon and Filetype Menus

As from Opus 5.5, each icon, and Name mode Lister if activated, has a special set of **Filetype pop-up menus**, accessed by pressing the RMB over the icon, or an entry in a Name mode Lister. Opus provides a limited set of common commands for these menus according to the type of icon or file. You can add your own favourite commands specific to icons and files of defined types.

When an icon is displayed for a file for which you have previously defined a Filetype, the Filetype menus are added to the pop-up menus for icons of this type. Select *Add* to define the Menu label and functions for your own custom menus.

Default Icon and Select Icon

The *Select Icon* button allows you to provide an icon image which Opus 5 will use when displaying this specific Filetype in either an Icon or Icon Action mode Lister, or when left-out on the Opus 5 Main Window. To add your own special image, either select the icon image required through the file requester, or drag and drop a suitable icon into the display area above this button. (We provide a special *DOpus5:Icons* directory to store such special icons.)



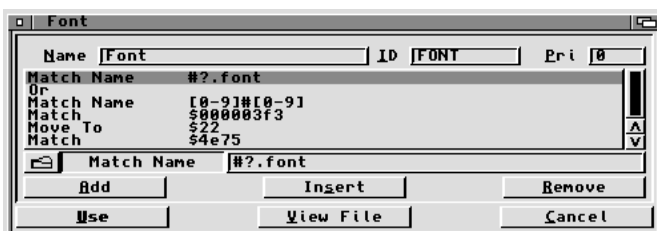
If an icon image is supplied, Opus 5 will use this defined icon for the internal AddIcon command instead of the system default.

Edit Class

Underneath the event list is a button which lets you modify the class or definition for this Filetype as discussed below.

Definition of a Filetype

Selecting *Edit Class* from the Event Requester brings up the File Class editor. Here, you specify the elements which Opus 5 will look for to recognise a specific Filetype.



Name : This is for the name of the Filetype.

ID: The ID will appear beside the Filetype in the Filetype manager screen. It is a shorthand way for Opus 5 to display the name for the Filetype definition.

Pri: The priority determines the order of matching different Filetypes against the file in question. Generally, it should be left at zero, but at times it can be very useful or even necessary to have different priorities.

For example, in the case of having two Filetypes defined, where one is a subset of the other (e.g., 24 bit ILBM pictures versus regular IFF ILBM picture Filetype), you would want the 24 bit IFF ILBM pictures to come first, because they are a special case of the regular ILBM picture Filetype. Otherwise, pictures (in this case) will be matched with the regular IFF ILBM picture Filetype and will never have a chance to match with the subset 24 bit ILBM picture Filetype. In this case, you

would set the Priority of the 24 bit IFF ILBM Filetype to a higher priority.

Underneath these fields is the actual script used to perform the Filetype identification. This is a series of actions that Opus 5 will perform in order to identify a file of a particular type. The action may be as simple as matching a filename to a pattern, or as complex as scanning an IFF form looking for data in a specific IFF chunk.

Below the list is a folder gadget, a read only field, and an argument field. These are used for editing the file identification definition as discussed below.

Add: Creates a new entry in the file definition script.

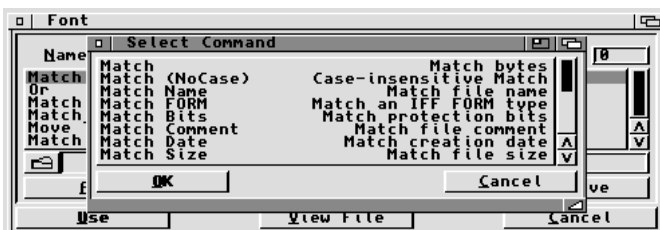
Insert: Inserts a new entry in the file definition script above the highlighted entry.

Remove: Removes the highlighted entry.

View File: Allows you to select a file to be viewed via the SmartRead command to help you edit the directives used to match the Filetype.

Editing the Filetype Definition

To edit a line, simply click on it and the read only and argument fields will be filled in. To change the command in the read only field, click on the list button and a list of other commands will be displayed. Select the one you want or press *Cancel* to abort.



The structure of the Filetype definition script consists of a clause or a sequence of clauses that describe what should be considered a matching file for a given Filetype. There are only two directives that delimit clauses: **And** and **Or**. These define what to do if a clause fails or succeeds. When all the clauses are finished and the result is true then the file is of the right type.

Edit Commands

- **And**

If the preceding clause succeeds, then also do the following clause; otherwise skip to the next clause. If the preceding clause failed, then execution stops and the file does not match.

- **Or**

If the preceding clause fails, then do the following clause; otherwise skip to the next clause.

Testing Directives

- **Match** text, \$hex, %binary or character '\xxx'

States that a sequence of bytes starting at the current file offset must match the given pattern. Match also supports binary matching. To match a single unknown character when text is given, use the '?' character. To match a single unknown byte when \$hex is given, use two of them (??). You can also use '\000' syntax in text to specify ASCII characters by their decimal number. A '\009' would be a tab character, a '\114' would be the lower case 'r'. Use '\063' to match a literal question mark. Examples:-

Match \$000003F3 (executables start with these bytes)
Match FORM????ILBM (the way a IFF ILBM picture starts)
Match Hey\009Overthere ("Hey", a tab then "Overthere")
Match \$FFFA (match hex characters)
Match %10110 (match bits)
Match text\127 (match 'text' plus a del character)

- **Match (NoCase)**

Is identical to the Match command but case-insensitive for ASCII matching (hex, binary and \xxx codes are still case sensitive).

- **Match Bits** +/- HSPARWED

Tests the file's protection bits. To see if a bit is set, put a + before the character. To see if the bit is unset use a -.

Example:-

Match Bits +RW (read & write must be on, others don't matter.)

Match Bits -E (executable must be off.)

Match Bits +RW -E (read & write must be on with executable off.)

- **Match Comment** text

Compare the supplied text string against the comment of the file. Any valid AmigaDOS wildcard pattern is usable here.

Examples:-

Match Comment Silly_Picture
(a file with 'silly_picture' as a comment)

Match Comment #?smeg#?
(any file with 'smeg' in its comment field)

- **Match Date** dates

Tests the date of the file against a given date. (See the *Select* command on page 122 for information about date strings and ranges.) Example:-

Match Date 08Sept95

Match Date < 10Jan96

- **Match Name** filename

Matches the given Name pattern against the filename. Any valid AmigaDOS wildcard pattern is usable here. Example:-

Match Name #?.ilbm

Match Name *.lzh

- **Match Size** > or < or = integer

Tests the size of the file against a value. Example:-

Match Size > 1000

- **Match FORM**

Performs a match for an IFF FORM type. Examples

Match FORM ILBM (match an IFF ILBM picture)

Match FORM SMUS (match an SMUS music file)

- **Match DT Group**

Matches a standard datatype group. This is only available for OS3.0 and above and the Amiga datatypes.library must have been installed on your system for this to function correctly. Examples:-

Match DT Group picture (any picture file known by datatypes, only first 4 characters are significant)

Match DT Group sound (match any sound file)

For further information, see the Amiga Documentation in datatypes/datypes.h for a complete list of current groups.

- **Match DT ID**

Matches a datatypes ID. The datatypes.library must have been installed on your Amiga for this to function correctly.

Example:-

Match DT ID jpeg (match a JPEG file)

This is dependant on what datatypes you have in your system.

- **Directory**

Matches directories. This command takes no further parameters. Example:-

Directory (match a directory)

Movement Directives

- **Move To** byte location (in decimal or \$hex)

Moves to a specific byte offset from the beginning of the file. Initially you are always at the beginning of the file, but you may have been moved in a previous clause, so you might want to put a MoveTo at the beginning of a clause in order to know exactly where you are. Examples:-

Move To 0 (back to beginning of the file)

Move To 100 (move to the 101st byte of the file)

- **Move** byte offset (in decimal or \$hex)

Moves to a byte relative to the current file offset. Examples:-

Move 16 (move sixteen bytes forward into the file)

Move 4 (move back four bytes from where we are)

- **Search For** text or \$hex

Searches (starting at the current file offset) for a certain byte pattern that matches the given pattern. See the Match command for valid options to use with this directive. If the match occurs, then the current file position will be the first character matched. Examples:-

Search For CMAP (look for 'CMAP', position on the 'C')

Search For M.K. (look for 'M.K.', position on the 'M')

A failure of any Movement directive means that the clause fails.

One example of usage is for the file class 24bit picture.

Example:-

Match FORM???ILBM (file must start with these characters)

And (if the previous cause is true then do the following)

SearchFor BMHD (then search for the BMHD chunk ID)

Move 16 (move sixteen bytes into the file)

Match \$18 (this must be 24 (or \$18 in hex) to be a 24bit picture.)

- **Find Chunk**

Searches for an IFF chunk. This command is similar to using Search For but much faster, since it understands IFF file format and skips non-matching chunks (instead of searching the whole file). It will also only match real chunk headers, whereas Search For is always likely to match on erroneous data in the file when searching for chunk headers. Example:-

Find Chunk BMHD (finds the next BMHD chunk)



We would suggest that you look at the predefined Filetype definitions to get an idea of the type of commands to use and what you can do with this system.

Extra Examples

Some other often used examples are

- a) To match a JPEG picture. (You must have jpeg datatype in your system.)

Match DT Group pict
Match DT ID jpeg

- b) To match a 24 bit ILBM picture

Match FORM ILBM
Find Chunk BMHD
Move 16
Match \$18

- c) To match a GPFax file in FAX IFF format

Match FORM FAXX
Or
Match FORM FAX3

- d) To match an AmigaGuide file

Match (NoCase) @database

The Automatic Filetype Creator

We realise that many users do not have time to carefully analyse files to determine their internal formats. In Opus 5.5, we have provided a system to automatically test and analyse files and create the Filetype definition for you. You can also have Opus call the Filetype creator whenever it fails to recognise a file. This is the simplest way to build up a list of usable Filetypes quickly. Unfortunately we cannot create the Command arguments or functions for you but we have come half way. The rest is up to you!

The Filetype module is designed to automatically find and create Filetypes for files or groups of files without the user having to be an expert on file formats. It provides two internal commands, ***FindFiletype*** and ***CreateFiletype***.

FindFiletype helps you quickly find which of your current Filetypes match a given file, whether they are actually installed in the DOpus5:Filetypes or in the storage directory.

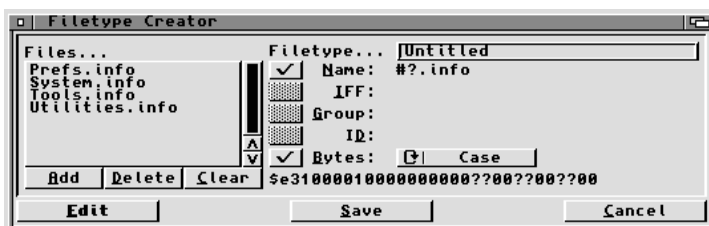


The display shows you a list of all the Filetypes that would match the given file in order of Filetype priority. The Filetype which Opus 5 would actually use will be highlighted and available for immediate editing. If a better Filetype is found in the storage directory than the one currently installed, the Filetype module can install it for you. If no Filetype is found in either the Filetype directory or the storage directory it will help you create a new Filetype by invoking the *CreateFiletype* command. It also allows you to edit the current matching Filetype.



The Filetype Finder only allows you to edit the current matching Filetype, not all the Filetypes shown in the list. Remember, the Filetype Finder is designed as a simple interface to help you rapidly indentify unknown files. For more complicated situations or where you wish to edit multiple Filetypes, you should use the Filetypes Editor directly.

CreateFiletype helps you to create a new Filetype for a single file or, preferably, from a list of files of the same type. It is designed to accept a list of files of similar type, analyse them for common Opus file matching elements and create a matching Filetype. The matching elements are displayed on the right of the display and you may select which specific elements to use to define the Filetype. If no items show up on the right then the AFC was unable to find common elements within the selected files. In such cases, you should selectively delete files from the list until it is possible to come up with a sensible Filetypes.



From the display, buttons allow you to add and remove files from this list. You can also simply drag and drop extra files into the list.

To the right of the file list is a series of checkboxes. Each one represents a standard Opus Filetype matching function. The functions are:

Name	Match Name	Match a filename
IFF	Match FORM	Match an IFF FORM type
Group	Match DT Group	Match a DataTypes group
ID	Match DT ID	Match a DataTypes ID
Bytes	Match	Match bytes
	Match (NoCase)	Case-insensitive Match

Name: This field shows the trailing part of the names of the files which match for all selected files. This will usually be the filename extension such as .gif or .info.

IFF: If all selected files are IFF FORM files and they all have the same FORM type, this type will be shown here. For example, ILBM or 8SVX

Group: If the Amiga datatypes system is available on your machine and all selected files are in the same datatype group, the group will be shown in this field. For example, 'text' or 'pict'.

ID: If the Amiga datatypes system is available on your machine and all selected files share the same datatype ID, it will be shown in this area. For example, 'amig' or 'ilbm' or any custom datatypes you have installed.

Bytes: This field will show up to the first sixteen bytes of the selected files using simple pattern matching. The cycle gadget allows you to enable or disable case-sensitivity.

Only the properties shared by ALL selected files will be shown in the display. If a file is added to the list which is not of the same type as those already selected, deleting that file from the list will update the fields automatically.

Any of these fields may be turned on or off individually.

The **Edit** button will bring up the standard *Opus Filetype Editor* enabling you to customise the match functions as well as adding default actions, pop-up menus and a default icon. After choosing the relevant fields by using the checkboxes, you may edit the Filetype further by hand by clicking on the *Edit* button.

If you decide to add or remove files from the listview after editing your Filetype, your edited changes will be lost. A requester will warn you if this is going to happen.

When a new Filetype is saved, Opus will recognise and use it immediately.



You must use the AFC with some thought! It is quite easy to create a Filetype which is far too general, which matches almost everything. After you have created a Filetype, test it with files of different types to ensure that you have made useful choices.

It is generally not a good idea to use the AFC to make a new Filetype based on a single sample file only.

This page is intentionally left blank.



Chapter Thirteen

The Opus Editors

Directory Opus provides a number of specialist editors and requesters to allow you to customise various objects such as buttons, Toolbars, menus etc. With Opus 5.5, these have been completely revised and many features have been added. Drag and drop support has been extended and more clipboard support has been added. Most of the editors now have extra menus to add some of the extra but less used functions.

The Function Editor

At the heart of Opus lies the power to configure your own commands or functions. An internal command or function tells Opus what to do when an event happens, such as a mouse click. Functions can be attached to any of the custom buttons, toolbar buttons, Filetype actions, menus, Hotkeys and Scripts. Each of these has its own configuration requester with unique aspects, for example, the title and colours used on a custom button.

The unique aspects of the various editors are described in turn in the following sections, but they all have a common portion, referred to as the *Function Editor*, which allows you to edit the Function attached to the specific object, button, menu and so on.

When in doubt, try drag and drop within Function Editors; either to swap function lines around within the one editor or to copy function lines to another editor. If you hold down shift while you drag a function line from one editor to another, the entire function will be copied.

The Function Editor



At the top of display is the *Function List* which shows the commands associated with this function. Beneath this list is a cycle button so you can select the function type and below this are buttons to allow you to modify the order and effectiveness of these commands.

- Add:** Adds a new Command to the end list.
- Insert:** Inserts a new Command at the highlight.
- Delete:** Deletes the highlighted line from the list.

To edit a line in the function list, simply click on it. The Command string will be copied to the field below the list for you to edit it.

Function Editor Menus

For Opus 5.5, a new set of menus provide extended clipboard support allowing you to Cut, Copy and Paste to and from the clipboard plus export the command lines as ASCII to a file.

Edit Fields

Below the Command list is a group of editing tools. These tools allow you to edit active Function entries. When you click on an entry in the Command List it becomes active, or an empty one is created when you select Add or Insert.

Command Type

The cycle button immediately beneath the function list allows you to specify which kind of Function is to be used. When you click on this button, it will cycle through the following types:-

Command Script	AmigaDOS ARexx	Workbench
-------------------	-------------------	-----------

Each of these functions is described below.

When you click on the small folder button just to the right of the Command Type button, a requester appears allowing you to pick an appropriate entry for the selected function type. Each of the following descriptions indicates the kind of requester which will appear.

Command: These are internal commands, built into Directory Opus. Many of these Commands can take parameters from buttons and menus as well as from ARexx. Internal Commands are documented in the Chapter Eight. The folder button brings up a list of internal commands.

AmigaDOS: Represents normal AmigaDOS programs. Such executables are launched as if you were running them manually from the CLI. Thus, with an output window enabled, they can receive keyboard input from the user and display output on the screen. The folder button brings up a file requester from which you may select the full command path to the application program.

Workbench: Workbench programs are also executable programs. However, they are launched as if you were double-clicking on their icons from Workbench. This can be an advantage, since many programs do not take arguments, or do not work at all if run from the CLI.

If the selected Workbench program is a *tool* (i.e., an executable program), Opus will look for its icon file to determine the necessary stack size to give to the program. If the icon cannot be located, Opus 5 will use a default stack size.

If the selected Workbench program is a *project* (a non-executable file created by another program), Opus will look for its icon file to find its *Default Tool*, the actual program needed to load the file. If the icon cannot be found, or a *Default Tool* can not be loaded successfully, Directory Opus will not launch the file. The project's icon is also used to determine stack size.

Workbench programs can also take arguments from Opus using the **{f}** and similar sequences. This can be very useful. DeluxePaint, for instance, does not accept arguments if run from the CLI; therefore you would be unable to select a picture file for DeluxePaint to load from Opus if you were running it as an executable.

If, however, you have the command defined as:

DPaint {f}

and have the Command type set to 'Workbench', DeluxePaint will be run as a Workbench program. From the Workbench, DeluxePaint will accept arguments, so the first file you selected would be loaded into DPaint automatically. The folder button brings up a file requester.

Script: Script files, also called batch files, are files that you might run with the Run command, or with the DOS Execute command from the CLI. Selecting a Function type as Script will cause the file to be executed as a script file. The folder button brings up a file requester which is initially set to S: because this is where script files usually reside by default.

ARexx: This type indicates that the Function is an ARexx script. The script will only be launched if ARexx is active in the system. The folder button brings up a file requester which initially shows the DOpus5:Arexx directory where most ARexx examples for Opus should be stored.



The address of the ARexx port is NOT set automatically. Scripts should use the ARexx ADDRESS instruction to address the command correctly to Directory Opus.

{}

This button is located next to the Function Edit Field. For all but internal commands, it displays a list of the arguments variables with a brief description of each variable. Function strings can contain many different command sequences to do different things with files and directories. The definition of each of these variables is discussed in detail below on page 163.

For internal commands, this button displays the template or options for this command. For example, for the *Play* command you would see a list with 'NAME' and 'QUIET/S'. For details of the different command templates, please refer to the specific command in the Chapter Eight.

Flags

Below the Command list is the Flags list. This is a list of all the flags available for custom Commands. These flags apply to all Commands in the Command list.



If the optional command arguments, such as {s!} etc, are used, setting some flags may cause the command to fail. See Special Arguments below.

The flags are:-

CD destination: If this is turned on, the current directory of the custom function will be set to the current DEST directory.

CD source: If this is turned on, the current directory of the custom function will be sent to the current SRCE directory.

Do all files: This causes the function to act on each selected entry in turn, instead of just the first entry. It is used for commands which do not support multiple filenames on the command line, where {F} to send all selected entries, would not work.

No file quote: This option enables Directory Opus to operate correctly with some older or poorly written software. Normally, whenever Opus sends a filename to a custom Command with the flags such as {f}, {o}, etc., the filename is enclosed in quotation marks. This allows you to use filenames containing spaces with external programs. However, some software does not interpret the quotation marks correctly. If you find this is the case with any program, simply select the *No file quote* flag.

Output to reader: Redirects all output from the function to a temporary file in T: directory, which is then read via the Opus text reader. This allows you to read the output of a program thoroughly, and even to print it. Note that if you are sending output to a file, the function cannot receive input from the keyboard.

Output to window: Opens a window for output from these function commands. The window will open on the Opus screen with the size and position as set from the *Environment / Output Window* settings.

Recursive dirs: Allows the function access to files within subdirectories. Normally used whenever a {f} or {F} or similar sequence would result in the name of a directory being included in the same way as a file. In other words, the function would not act recursively on all files within the directory.

If this option is enabled, the names of all files within that directory, and within subdirectories within the directory, and so on, are included in the program's parameters. This allows the function to act on all files in the directory and not just on the directory itself.

Reload each file: Causes Opus to rescan a file after it has been acted upon by a function, and updates the size, datestamp, comment and protection bits of the file. You can therefore reflect changes in size, for instance, made by a text editor to a file.

Rescan dest: This flag makes Opus reload the destination directory when the function terminates. This, and the option below, allows Opus to display correctly any changes made to either directory window by external programs, such as archivers.

Rescan source: Makes Opus reload the source directory when the function terminates.

Run asynchronously: Indicates that the functions are to be launched as a new process, and Directory Opus is not to wait for it to return.

Window close button: Tells Opus to wait until you click on the output window close button before closing any output window.

Window on Workbench: Tells Opus to open an output window (if one is required) on the Workbench screen (actually the default Public screen), instead of on the Opus custom screen. If the Opus screen has been set to be the Default Pubscreen, this setting becomes irrelevant.

Key

Allows you to set a shortcut or hot key sequence to activate this function. Pressing the shortcut key will act exactly as if you selected this function from a button, menu or with other action. You may use any combination of *Shift*, *Alt*, *Ctrl*, or *Amiga* qualifiers *plus any key*. With Opus 5.5, key sequences are automatically recognised. Just press the key combination, for example 'left shift + left Amiga + A' and Opus will insert the correct commands for 'lshift lcommand a'.

If for some reason you wish to enter your own key sequence, lock the *Caps Lock* key down first, then enter the desired command string. If you wish to delete a previously entered hot key, you will need to depress the *Caps Lock* key to enter string mode first then delete the entry for the key.

Argument Variables {}

AmigaDOS, ARexx and other external commands launched from Directory Opus 5 support control sequences to insert special information in the command line. These control sequences are in the form of codes enclosed within braces. These codes can appear anywhere in the command line, and will be replaced with the appropriate information when the command is executed. If the codes cannot be parsed correctly, Opus may refuse to execute the command. ***The variable names are case sensitive.***

The variables are as follows:-

{f} First selected entry (with path)

Insert the path and filename of the first selected entry. The entry will then be deselected. If you specify a minus sign (-) after the letter "f", the filename will be stripped of any suffix it has. For example,

AmigaDOS Rename {fu} {f-}.lzx

would replace any suffix the selected file had with a .lzx suffix. That is, "ram:test.lha" would become "ram:test.lzx".

{fu} First entry (with path, do not deselect)

The same as {f} except that the entry is not deselected after its name is used. This allows you to use the same filename more than once in a command invocation. The {fu} command may be used multiple times, and should be followed by a {f} to signify that you are done with this file. This command also supports a minus (-) sign to strip any filename suffix.

{F} All selected entries (with paths)

Insert the path and filenames of all selected entries. The entries will then be deselected.

There is a nominal limit of 512 characters for each command line. Therefore, if the length of a command exceeds 512 characters because of a {F} sequence (or indeed for any reason), another command line will be created for the next group of entries, and so on until there are no remaining selected files.

{o} First selected entry (name only)

The same as {f} except that only the filename is used, not the complete pathname. This command also supports a minus (-) sign to strip any filename suffix.

{ou} First entry (name only, do not deselect)

The same as {fu} except that only the names of the selected entry is used, not the complete pathname. It is quite ok to intermix {fu} and {ou}, and follow with a {f} or a {o}. This command also supports a minus ('-') sign to strip any filename suffix.

{O} All selected entries (names only)

The same as {F} except that only the names of the selected entries are used, not the complete pathnames.

{d} Destination directory window's path

This will insert the path of the destination directory window. Only one destination is supported and any *locked* destinations are ignored.

{s} Source directory window's path

This will insert the path of the source directory window. Only one source is supported and any *locked* sources will be ignored.

{v} Environment variable

This will insert the value of a global environment variable in the command line. The format of this command is {v<varname>}, where <varname> is the name of the environment variable to insert. For example:-

```
AmigaDos echo Using Kickstart {vKickstart}
```

{Rd} Directory requester

{Rf} File requester

{RF} File requester in save mode

These three commands allow you to access the Amiga file requester from a function to accept the name of a directory or full path for a filename as required.

{Qa} Query argument

This allows you to access an argument string from selected functions.

For *Disk inserted* and *Disk removed* scripts, you could have the following function for the Disk inserted script :

```
Command Confirm Disk inserted in {Qa}. Read it?  
Command Scandir new {Qa}
```

Opus knows about CrossDOS, and so will return PCx: instead of DFx: if an MS-Dos disk is inserted. For disk removed, DFx: is always returned. Only diskchange on the four floppy drives is supported. For *Bad disk inserted* scripts when you insert a non-DOS (and non-MSDOS if CrossDos is running) disk in any floppy, you might want a function such as :

```
Command Confirm Non-DOS disk inserted in {Qa}. Format?  
Command Format {Qa}
```

The lister handle of a newly opened lister or the name of a button bank or group is available to script functions in {Qa}.

{Qd} Query destination lister

This will insert the handle of the destination lister.

{Ql} Query source lister

This will insert the handle of the source lister.

{Qs} Query screen name

This will insert the name of the public screen Opus 5 is currently open on. This will usually be DOPUS.1, but it may be DOPUS.2, Workbench, or indeed any other public screen you have configured Opus to open on. Use this to make external programs open their window on the Opus screen (without having to turn on the Default PubScreen option). For example:-

```
AmigaDOS sys:tools/calculator pubscreen={Qs}
```

{Qp} Query port name

This will insert the name of the Opus 5 ARexx port. This will usually be DOPUS.1, but may be DOPUS.2, DOPUS.3, etc, depending on the number of copies of Opus you have running at once. Note that while the ARexx port name may be the same as the public screen name, it is quite possible for them to be different. You will want to use this command whenever you launch ARexx scripts, to pass the script the name of the port it must communicate with. For example:-

```
ARexx DOpus5:arexx/myscript.dopus5 {Qp}
```

{Rs} Request String

This will display a requester to the user asking for text input. The complete format of this command is *{Rs}<title>:<default>}*. *<title>* is the optional title of the requester, and *<default>* is the optional default string to display. Either of these can be omitted (if you specify *<default>* you must precede it with a colon).

The *<default>* string can contain control sequences of its own. These are:-

- [o]** inserts the last filename used by a {f}, {o}, etc
- [f]** inserts the last used filename, including the full path
- [s]** inserts the source directory window's path
- [d]** inserts the destination directory window's path

For example, it would be possible to have your own custom rename button with the following function:-

```
AmigaDOS Rename {f} {RsEnter new filename:[o]}
```

In this case a string requester would open with the title "Enter new filename" and the default value of the string field would be the original name of the file.

Optional Arguments

Option arguments are a new feature of Opus 5.5 which allow variable argument to fail gracefully if there are no selected files and/or no SRCE or DEST arguments.

With normal arguments, if the arguments cannot be parsed correctly the command will fail and Opus will not execute the command script at all. This is done for safety. It can be very dangerous if you have used destructive commands, such as *Delete*, and have incorrectly given the command the wrong arguments.

However, there are many cases when you may wish to call an AmigaDos program with or without arguments. With Opus 5.5 we have added a special exclamation mark qualifier. If this follows the argument variable then the argument will be treated as optional and the command will be executed even if the variable parsing fails. These argument variables are

- {s!}** - uses source path if present, but doesn't require it
- {d!}** - same for destination
- {f!}** - first file if present, but not required
- {o!}**
- {F!}** - etc...

Example 1: A button to run DPaint with a filename if one is selected, or without a filename if not..

Workbench Apps:Gfx/DPaint {f!}

Using the old {f} command, the function would only be launched if there was a selected file.

Example 2: To call CygnusED with or without a file selected, use

AmigaDOS ED {f!}



Note that if these optional arguments are used, it is sill possible for the command parsing to fail because of an incorrectly set flag. With Example 2, if you had set the 'CD Source' flag and did not have a valid SRCE Lister, the command would fail before Opus even looked at the argument variable.

A Simple Example

The following is an example of a relatively simple function. It is the function from the default configuration that allows you to create LHA archives. The {Rs} sequence opens a requester asking the user to enter the name of the archive to create. The {d} sequence in front of this causes the path of the destination to be prepended to the requested filename. The final {O} causes the filenames of all selected files to be added to the list for archiving. Because the *CD source* flag is turned on, {O} can be used instead of {F} (since full pathnames are not required). This allows more filenames to be added per line (remember the maximum line length of 512 characters), and will result in the LHA program having to be invoked fewer times.

Function definition:

```
AmigaDOS lha -e -x a {d}{RsEnter archive name} {O}
```

Flags turned on are:-

CD source, Output to window, Recursive dirs,
Run asynchronously

An example output of this command:-

```
cd "work:"  
lha -e -x a "ram:test.lha" "file1.c" "file2.c" "foo/file3.c"
```

A Complex Example

The following is an example of a complex function. It demonstrates the power and flexibility that the argument parameter codes make available to you.

This function will automatically convert any selected LHA archives into LZX archives. The .LHA suffix is removed automatically from the filename and replaced with a .LZX suffix. The original LHA archive is first extracted into a temporary directory in T:. It is then re-archived using LZX. The original LHA archive and the extracted files in the T: directory are then deleted. All selected

The Function Editor

LHA archives are processed in turn, due to the *Do all files* flag being turned on. At the end of the process, the source directory is rescanned automatically (the *Rescan source* flag).

Of course, you need to have both the LHA and LZX programs in your path for this function to work.

Function definition :

```
AmigaDOS  mkdir t:lxztemp
AmigaDOS  lha -x -M x {fu} t:lxztemp/
AmigaDOS  cd t:lxztemp
AmigaDOS  lzx -e -x -r a {fu-}.lzx #?
AmigaDOS  delete {f} quiet force
AmigaDOS  cd ram:
AmigaDOS  delete t:lxztemp all quiet force
```

Flags turned on :

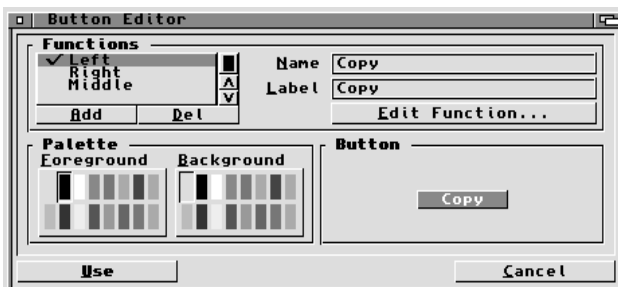
Do all files, Output to window, Rescan source

An example output of this command :

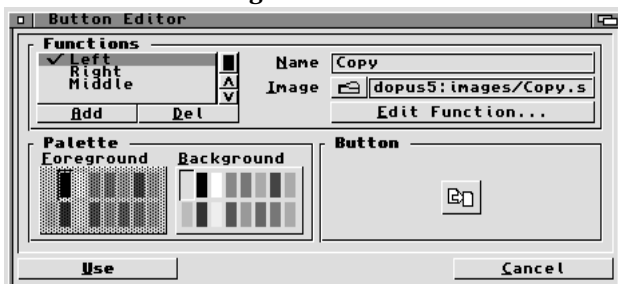
```
mkdir t:lxztemp
lha -x -M x "dload:foobarbaz.lha" t:lxztemp/
cd t:lxztemp
lzx -e -x -r a "dload:foobarbaz.lzx" #?
delete "dload:foobarbaz.lha" quiet force
cd ram:
delete t:lxztemp all quiet force
```

The Button Editor

The button editor is displayed whenever you wish to edit a button in a custom Button Bank or in a ToolBar. From the Button Bank editor, either double-click the desired button, or highlight it and select 'Edit' to display the Button Editor.



Editing a Text Button



Editing an Image Button

The Button Editor allows you to change the following features of a button:-

Functions: The listview gadget gives you the choice of setting a function to be activated by the Left, Right or Middle mouse buttons and, for button banks, additional pop-up menu items. Any function which is actually defined will have a check mark displayed next to it.

Add: For button banks only, you may add any number of additional pop-up menu items to any button. These functions

will initially be named '*new function*', but you should give them a meaningful name which will appear in the pop-up menu.

Del: For the left, right, and middle mouse button functions, this will delete the function and related items from this entry. For pop-up menu entries in button banks, the entry itself will also be deleted from the listview.



You can use drag and drop to rearrange the entries in the listview, and even to copy an entry to another Button Editor, Menu Editor, or Function Editor.

Name: When editing a custom button, enter the name you wish to appear for the button. The name you choose for the left mouse button will be the initially displayed name on the button, but you may also set different names for each mouse action as well as for each pop-up selection.

Image: When editing a graphical button bank or the Lister Toolbar, you select the file which contains the image for the button. Clicking on the list button, immediately to the left of the word Image, will display a file requester. Select either the name of an IFF brush file or an icon file (.info file) and Opus will load the image and use it for the selected button. *Alternate select images* for graphical buttons are now supported. That is, if you are using an icon for a button image, the selected imagery of the icon will be shown when you click on the button. For IFF images, you can supply an AnimBrush, the second frame of which will be used for the selected imagery. If the name of a file used as a button image ends in '*.noremap*', that image will not be colour re-mapped. Image buttons inherit their image from the left button function if they don't have an image explicitly defined. If the image specified for a button is not found, a default image will be used.



Hint: You should attempt to keep the images used for each button or toolbar image at approximately the same size. Opus will calculate the size of each button from the largest image.

Label: When editing a text button, this is the text that you will see printed on the button. The *Label* and *Name* fields will usually be the same. In fact, if you fill out one of these fields

when the other is empty, the text will appear in both fields automatically.

Edit Function: Selecting this button displays the *Function Editor* so you may add, change or delete the command function you wish to use with this button. See the *Function Editor* on page 157 for more details.

Palette: For button banks only, this displays a colour requester where you may set the colours for your buttons. The number of colours displayed in the colour requester will be determined by the depth of the screen you have chosen for Opus and the number of User colours you have defined in the Environment.

For **text buttons**, you may select a different foreground colour (the one used for the text) and background colour (the base colour of the button) for each mouse button type.

For **image buttons**, you may select the background colour only. The foreground colours come from the image and may only be changed by editing the image itself.

For **Toolbar buttons**, neither the foreground nor background colours may be set.

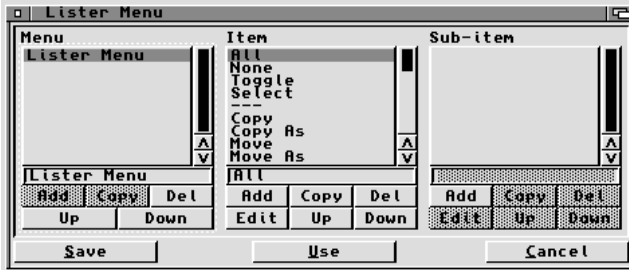
Button: This area shows you the image or label of the selected function. You can drop an IFF brush file or an icon file here to add an image to the function if you are editing a graphic button. You can also copy the whole button by dragging this image or label onto another button, button editor, menu editor etc.

Use: Accept any changes you have made to this button

Cancel: Cancel any changes you made to this button.

The Menu Editor

The Menu Editor is invoked when you wish to edit the Lister Menu or the global User Menus.



The three lists represent the various levels for the menus, the menu items, and the sub-items respectively. Only one *Lister Menu* is allowed, but you may define any number of *User Menus* that will fit on your screen. Beneath each list there are a series of buttons:-

Add: Adds a blank menu item at the end of the list. If you hold down SHIFT when pressing *Add*, the new item will be added at the current position instead of at the end of the list.

Copy: Creates a new entry identical to the highlighted item and adds it to the end of the list. If you hold down SHIFT when pressing *Copy*, the new item will be added at the current position instead of at the end of the list.

Del: Removes the highlighted item from the list and moves the remaining items up one place.

Edit: Displays the *Function Editor* and allows you to edit the functions attached to this menu item.

Up: Moves the highlighted item up one place.

Down: Moves the highlighted item down one place.



To edit any entry quickly, just double-click on it.

As well as these buttons, there are some extra controls for the menu editor:-

Drag & drop moves the item.

Drag & drop with **shift** held down copies the item.

Tab/Shift Tab to change to the next/previous active list as indicated by a dotted rectangle.

Cursor up/down to change selection in active list.

SHIFT + Cursor up/down to move the selected item up or down.

Return to edit the selected item in the active list.

+ to add an item to the current listview.

DEL to delete current item from the listview.



Drag and drop is very powerful in the menu editor. You can use it to rearrange items in each list, to copy an item from one list to another, to copy an item to a different Menu Editor, or even to different editors such as the Button, Hotkey, Script, and Function Editors.

Adding Menu Separators

When creating menus, it is often a good idea to visually separate the menus into groups of related items. This makes reading a menu list much easier. Traditionally in the Amiga, we use a special flag called `NM_BARLABEL` to tell the Amiga OS to put in a special separator bar. If you wish to add these separators to your custom menus, simply put in a row of minus signs, minimum of three such as '---', and Directory Opus will interpret this as an instruction to place a separator at this position in the menu list.

Menu Hot Keys

If you have a hot key set for any User Menu item of rcommand plus key, or rcommand plus shift plus key, the hot key will be displayed in the menu using the standard amiga menu symbol. Note that letters can only be uppercase, and the case is actually ignored when you press the key. This is an Amiga OS limitation, not Opus.

The Menu Editor Menus

Just as with the other editors in Opus 5, there are extra options provided by menus. These are:-

The Project Menu

New: Creates a new blank menu list.

Open: Displays a file requester allowing you to select and load a new set of menus.

When first run, Opus uses a default filename for both the *Lister Menu* and the *User Menus*. Once you load a new set of menus using a different name, this new name will be kept and used internally as the reference to that set of menus. If you subsequently save the particular set of menus, it will be saved under this name unless you use the *SaveAS* option. If you save the Environment, this 'new' filename will be stored.

Save: Saves the displayed set of menus to disk under the current name.

SaveAs: Saves the current set of menus but allows you to select a new filename.

Quit: Same as Cancel above.

The Edit Menu

Reset to Default: Attempts to reset the particular set of menus to the default set as defined when you installed Directory Opus 5. Because there are many settings, these defaults are not actually built-in to Opus 5, instead Opus will look for and load special default files. For correct operation of Opus, you should never overwrite any files in the Buttons drawer with the name ending in '_default'. Otherwise you will have to reinstall Opus to recover the default settings.

Last Saved: Reloads the last saved set of menus and resets the display.

Restore: Restores the displayed menus to the state when you first opened the Menu Editor.

The Button Bank Editor

Directory Opus 5 allows you to create any number of custom Button Banks and Lister Toolbars containing your favourite commands. You create and edit these custom buttons from the *Button Bank Editor*.

The Button Bank editor will appear automatically when you create a new button bank via the *Buttons/New* menu item. To edit an existing button bank, select it, then choose the *Buttons/Edit* menu item, or choose *Edit* from the button bank's pop-up menu. You can also edit a button directly by holding down the *Alt* key and clicking on the button.

The current Lister Toolbar can be edited via the *Listers/Edit Lister Toolbar* menu item, or by holding down the *Alt* key and clicking the Toolbar button. With Opus 5.5, Toolbars are now edited with the normal Button Bank Editor as with normal buttons, but with one or two restrictions as discussed below.



From this editor you can edit any button in any button bank on the screen. Although it is not generally a good idea, you can even edit multiple buttons at once. But this can be useful when you wish to compare the function commands you have assigned to different buttons.

A button bank is defined as either *text* or *graphical* buttons arranged in a series of rows or columns. Once you have created a bank, you may resize the window to display as many of the buttons as you wish. For *text* buttons, the button bank window can be resized to any horizontal width and Opus will stretch the

button width to fit the available columns. The vertical size is restricted by the height of the chosen font.

When using **Graphical** (or image) buttons, the horizontal and vertical size of the bank is limited by the sizes of the button images themselves.

When the Button Bank Editor is displayed, select the bank of buttons you wish to edit and the editor will display the details of that bank. Selecting a specific button will cause the button to flash to indicate the row, column and button being edited. A double-clicking on a button will invoke the *Button Editor* directly.

The Button Bank Editor is divided into three sections: Bank, Button, and Appearance.

The options presented in the bank section allow you to control the shape and font of the button bank as follows:-

Add: Adds another blank row or column to the selected button bank. New columns are added to the right-hand side of the current bank while new rows are added the bottom. When you add rows or columns, you may need to resize the window to reveal them.

Insert: Inserts a new blank row or column at the highlighted position.

Delete: Deletes the row or column underneath the highlighted button. **Care!** This will delete the complete row or column and all the details attached to the buttons therein.

Xform: A very special button! This function allows you to convert rows into columns and vice versa, while attempting to preserve the total number of buttons in the bank. It uses a simple integer method to swap buttons between rows and columns.

Font: Select the font and size to be used for the text in all buttons in the selected text button bank.

The Button Bank Editor

The Button section of the editor provides editing and clipboard operations.

Edit: Displays the *Button Editor* allowing you to set the command functions, colours and other parameters for the selected button.

Copy: Copies the highlighted button to the *Button Clipboard*.

Cut: Deletes the highlighted button from the bank and places it in the *Button Clipboard* for later reference.

Erase: Deletes the highlighted button from the button bank. The button is discarded and not stored in the clipboard as with the *Cut* action.

Paint Mode: A toggle switch which provides a quick method of setting the foreground (text buttons only) and background colours of buttons directly rather than through the button editor itself. When selected, a palette selector is displayed. Select the desired foreground or background colours and then click on a given button to change the colour directly.

Show Clipboard: Toggles the clipboard window on and off. The clipboard is always present and can be used via the above buttons, this switch merely provides a drag and drop interface to the clipboard.

When enabled, the clipboard is a small window with a scrolling button area. This is a temporary scratch pad for use while editing buttons. You may copy (or drag and drop) buttons to and from this temporary area.

Clear: This button on the clipboard is used to clear the clipboard of all temporary button definitions.

The Appearance section provides several new options for Opus 5.5 to allow you to customise the look and feel of each button bank.

Full Border: With this flag turned on, the button bank will use a normal window with close, zoom, depth, iconify, sizing, and scrolling gadgets. When turned off, the button bank uses a

compact, streamlined window with only a thin dragbar. The horizontal drag bar on these button banks is size-adjusted for the current screen mode, so it is half the height on a lo-res screen. This drag bar can also be used to bring the window to the front, by double-clicking on it. If you want to access the button banks' pop-up menu, you must have the pointer over this dragbar when you press the right mouse button. The only drawback with the new-look button banks is that since they have no scroll-bars, you cannot have more buttons than will fit on the screen.

Dragbar Orientation: With the *Full Border* option turned off you may set the dragbar to be a narrow horizontal bar at the top of the button bank, or a narrow vertical bar on the left side of the button bank. You can also set this to automatic, in which case Opus will determine whether a horizontal or vertical dragbar would be more compact.

Borderless Buttons: When you turn this option on, the button bank will be made more compact by removing the borders between each row and column of buttons. This may be particularly useful if you wish to use custom button images that have borders already drawn into them.

SimpleRefresh: This option changes the method used by Opus to refresh the button bank imagery if it has been damaged. Normally this is left up the Amiga graphics system. With SimpleRefresh, Opus will perform the refresh directly. SimpleRefresh uses less memory but may be slower depending on your system.

No 'Dog ears': Enable this option if you wish to suppress the 'dog ears' which are normally used to indicate that a button has extra functions attached to it for the right or middle mouse buttons, or for extra pop-up menu items.

Moving a Button Bank

Normally, to move a button bank to a new position on the screen, you simply use the window drag bar gadget in the Window title area. However, because this area also contains the close window, zoom, window depth, and iconify gadgets, it is possible to create a

button bank (especially if using small graphic imagery for the buttons) that is too narrow to permit access to all the title bar gadgets, especially the drag bar. For such cases, we have also added a window drag gadget on the very left-hand edge of each button bank. If you click and hold the left border, you will be able to drag the bank to the new position.

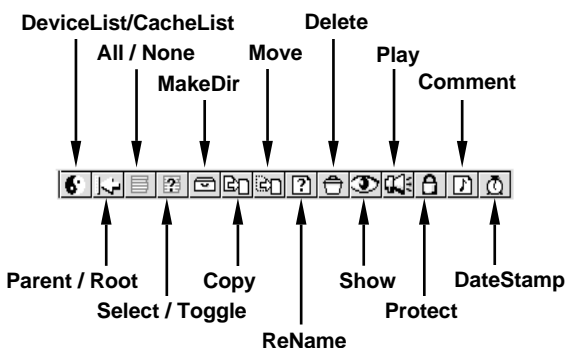
Editing the Lister ToolBar

As you have seen by now, the Opus 5 Listers may have an optional toolbar showing a series of small icons. Each of these icon images is actually a standard Opus 5 action button for which you may define separate actions for left, middle and right mouse clicks.

Since toolbars are just versions of Opus button banks, with Opus 5.5 the toolbars are edited with the standard *Button Editor*.

The Directory Opus 5 installation comes with a few sample toolbars for you to choose from but you can readily edit both the images and the actions as you desire. By default, Opus 5 will load the file '*Buttons/ToolBar*'. But, you may define your own toolbar if you desire. It is also possible to set a different toolbar in each Lister if desired. (See the *SET* command.)

Sample ToolBars



The sample toolbar which is supplied with Opus 5 is shown above with the associated functions attached to left and right mouse clicks respectively. You may easily edit these to your own requirements or create a complete new Toolbar of your own.

Editing the ToolBar

To edit the ToolBar, select '*Lister/Edit Lister Toolbar*' from the global menu or hold down the *Alt* key and click on a specific icon in the toolbar. The later action will allow you to edit a specific button immediately. This will display the *Button Editor* but provide some extra menus for the ToolBar. These are:-

The Project Menu

New: Creates a new blank toolbar.

Open: Displays a file requester allowing you to select and load a different toolbar configuration.

When first run, Opus uses a default file name for the lister toolbar. However, once you load a toolbar under a different name, this new name will be kept and used internally as the reference to the current toolbar. If you subsequently save the toolbar, it will be saved under this name unless you use the *SaveAs* option to change it. If you save the Environment, this 'new' filename will be stored and used next time you load Directory Opus.

Save: Saves the current toolbar to disk using the current filename.

SaveAs: Saves the current toolbar but allows you to select a new filename.

Quit: Same as Cancel above.

The Edit Menu

Reset to Default: Attempts to reset the toolbar to the default set as defined when you installed Directory Opus 5. Because there are many settings, these defaults are not actually built-in to Opus 5, instead Opus will look for and load a file called '*Buttons/Toolbar_Default*'. It will also reload the image files defined therein. For correct operation of Opus, you should

never save over this default file. Otherwise you will have to reinstall Opus to recover the default settings.

Last Saved: Reloads the last saved set of toolbar buttons and resets the display.

Restore: Restores the buttons to the state when you first opened the Lister ToolBar Editor.



Chapter Fourteen

Opus 5 Utility Requesters

The Diskcopy Requester

This function allows you to make an exact copy of one disk on another. When this function is called, a requester with several buttons appears.



From..: This list contains the possible disk drives that may be used as the source. When you click on one, it becomes the selected drive.

To..: This list contains the possible destination drives which are compatible with the selected source drive. The source disk drive is always available as a destination to allow you to make single drive copies. This only makes sense with removable media such as floppy drives as it would accomplish nothing to copy a hard drive to itself.

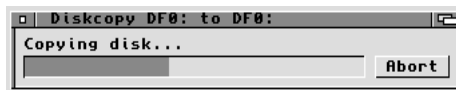
Utility Requesters

Verify: This button allows you to turn off the integrity verification when writing data to the destination drive. Although it is faster, you probably won't want to do this.

Bump Name: This button allows you to change the volume name using the same naming convention as Workbench's Diskcopy. (See AmigaDOS documentation for details.)

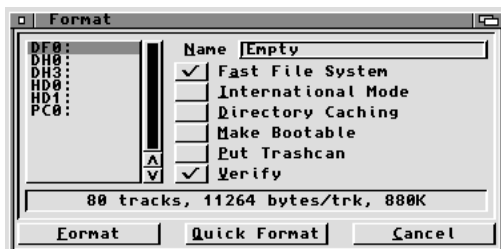
This function will not copy any protected software, or non-AmigaDOS format disks.

Selecting the '*Diskcopy*' button will start the copy. The '*Cancel*' button will abort without attempting to copy the disk.



The Format Requester

This allows you to format a new disk. All new disks need to be formatted before the computer can write to them.



When the *Format* command is called, a requester with several buttons appears. On the left side is a list containing the devices which can be formatted using this operation. The selected device is highlighted. Be sure the device you intend to format is the one that is highlighted!



Warning! *This option will destroy existing data on a disk. Be sure you want to erase the data before you click Format or Quick Format buttons.*

Name: This field allows you to give a volume name to the drive to be formatted.

Fast File System: This allows you to format a device using the Fast File System option of AmigaDOS. You should consult AmigaDOS documentation for more detail.

International Mode: This allows file and directory names to include accented characters.

Directory Caching: This will decrease the capacity of your disk but the directory reading speed will be much greater.

Put Trashcan: This button allows you to put a trashcan in the root directory of the newly formatted device.

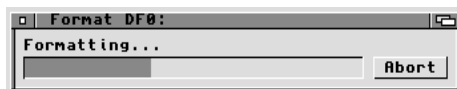
Make Bootable: If this button is selected, Opus 5 will install a standard AmigaDOS bootblock on the disk, making it bootable.

Verify: This button allows you to disable the format verification. As with the Diskcopy function, the process is faster with Verify turned off, but you won't be made aware of any errors, so it's better to leave Verify turned on unless you trust your disks completely (you really shouldn't).

Format: This button begins the formatting process. Be very careful that you have selected the correct device. Once a Format begins, it can be aborted, but data will be lost!

Quick Format: When this button is selected, the disk will be initialised (wiped). This provides an extremely fast way to erase an old disk. It will not work on new disks however, only on disks that have previously been formatted.

Cancel: This button will abort the process without attempting the format.



The Print Requester

This Requester gives you full print formatting control for text files.



These configuration options work in accordance with the Amiga Printer Preferences. The Amiga Preferences may override these preferences or simply make the output look silly. For example, you may not be able to use these options to display more lines on a page than is specified in Amiga Preferences. All printers are not created equal. Some printers will ignore some of these configuration options.



You may adjust the following configuration items:-

TEXT

Left Margin: This field contains the number of characters to skip before printing each line.

Right Margin: This field contains the number of printed characters allowed on each line. The Left Margin characters are not included in this value. For example, a Left Margin of 5 and Right Margin of 70 will result in the last printable character in the column 75.

Tab Size: This field contains the number of spaces to which a tab character is equivalent. Opus 5 converts tabs to spaces and will insert the appropriate number of spaces to create columns

based on Tab Size. For example, a Tab Size of 8 specifies Tab positions of 8, 16, 32, 40, 48, 56, and 64.

Quality: This button cycles between Letter, and Draft. Some printers can be toggled between Letter and Draft quality printing.

Pitch: This button cycles between Pica, Elite, and Fine. These values specify the size of letters to print. Your printer will determine the exact dimensions of these values.

Output: By default, the output will be sent to the current Preferences printer. However, you can redirect the output to a file of your choosing.

Printer: This option directs output to the printer.

File: When this option is enabled, output is directed to a selected disk file or device. When printing starts, a file requester is presented. Enter the file path required or enter the device name, such as PAR: or SER: etc.

Header/Footer

Configuration: This button cycles between Header and Footer. The Title, Date and Page no. buttons can be used when creating a Header or Footer line for each page in the print out. When the configuration button is Header, these buttons affect the Header line; otherwise, they affect the Footer line. By default, neither are created.

Title: When checked, a title will be generated. By default, the filename will be the title. However, you can override this by putting text in the Title field. You can have different titles in the header and footer lines.

Date: When checked, the current date will be printed. Usually this is enabled for either the header line or the footer line, but not for both.

Page No: When checked, the page number will printed. Usually this is enabled for either the header line or the footer line, but not for both.

Style: This button allows you to modify the appearance of all the printed text except the headers and footers. (Some printers do not support all of these styles.) Clicking on the Text Style Cycle button allows you to choose from the following options: Normal, Bold, Italics, and Unlined (Under Lined).

Print: When you click this button, Directory Opus 5 will begin printing the information.

This function will print all selected files, one at a time with a form feed sent to the printer between each file.

If you select only one file to print, the print routine will be started up as a separate process, allowing you to continue working with Directory Opus 5. To cancel this type of print, simply select the print function again. A requester will appear asking if you want to continue with the print or halt it. This requester will also appear if you attempt to quit Directory Opus 5 while a print operation is in place, as you cannot quit until the print has finished.

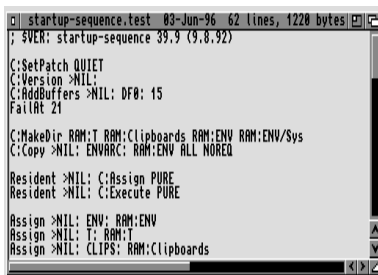
Even if you abort a print, the printer may not actually stop for some time. This is because most printers have buffers, some quite large ones, which store data for printing and will need to empty themselves before the print out will stop.

Cancel: This button will abort without attempting to print.

The Opus 5 Viewer

When required, Directory Opus 5 uses its own in-built program to display selected files in ASCII, ANSI, or HEXadecimal formats.

The viewer opens as a separate window either on the Opus 5 screen or on its own screen. The viewer's display is fully buffered so you may scroll backwards and forwards in the file as required, using the scroll bars or the keys. You may re-size the window as required with the size gadget on the bottom right.



The screenshot shows a window titled 'startup-sequence.test 03-Jun-96 62 lines, 1220 bytes'. The content of the file is as follows:

```
; SVER: startup-sequence 39.9 (9.8.92)

C:SetPatch QUIET
C:Version >NIL:
C:AddBuffers >NIL: DF0: 15
FailAt 21

C:MakeDir: RAM:T: RAM:Clipboards: RAM:ENV: RAM:ENV/Sys
C:Copy >NIL: ENVARC: RAM:ENV: ALL: NOREQ

Resident >NIL: C:Assign: PURE
Resident >NIL: C:Execute: PURE

Assign >NIL: ENV: RAM:ENV
Assign >NIL: T: RAM:T
Assign >NIL: CLIPS: RAM:Clipboards
```

The name of the current file is displayed in the window title bar, along with various details about the creation date and size of the file being displayed. The following keys are active:-

up or down arrows	Move up or down a line
U	Moves up a page
D	Moves down a page at a time
Cursor keys	Move up or down a line
Cursor key + Shift	Moves one page
T or Cursor up + Ctrl	Moves to the top
B or Cursor down + Ctrl	Moves to the bottom
Q or Esc	Exits the viewer
Space	Page Down
Backspace	Page Up
P	Print file
S	Search
N	Next search
Right Amiga + C	Copy to clipboard

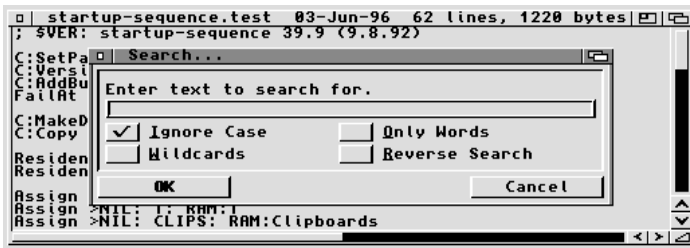
The arrow keys and **PgUp**, **PgDn**, **Home**, **End** are also available.

The Viewer Menus

The File Menu

Next: If you have selected more than one file, the next one will be read when you exit. To exit without reading the next file, press the 'Quit' button.

Search: Searches for a string. Limited pattern matching is provided by using the '?' character.



Repeat Search: Continue the search from the current position using the same search pattern.

Print: Prints the current displayed file.

Save As: Saves a copy of the file using the standard ASL file requester.

To Editor: Lets you edit the file with your preferred text editor as specified in the *Settings/Pick Editor* menu item.

Quit: Quits the viewer.

The Settings Menu

Tab Size: Allows you to specify how many spaces to be used in place of any TAB (\$09) characters found.

Mode: Choose to display the file in either normal, ANSI, or Hexadecimal mode.

ANSI mode: Some text files may contain imbedded ANSI sequences to provide extended formatting of the text using Bold, Italics and other sequences. When set in this ANSI mode from the menu, the Viewer is capable of displaying most of the standard ANSI sequences.

Hex mode: The viewer can display files in hexadecimal format. This allows you to view binary files and other files containing non-textual characters. The file is displayed in the following manner: The first value is the offset, displayed in hex. This is the number of bytes you are into the file. The next four values are each a four byte longword, with the actual ASCII representation at the end. Any non-text characters are shown as a '.' character.

Use Screen: Toggles whether the Viewer is opened on the Opus 5 screen or its own screen.

Pick Screen Mode: Allows you to choose the screen mode to use for the Viewer.

Pick Font: Allows you to choose the font to be used to display the text in the Viewer.

Pick Editor: Allows you to choose the command line used to invoke an external text editor. The "%s" will be replaced with the name of the file. For example:-

Ed "%s"

This will start a text editor called 'ed' if you have one. The double quotes are there so that filenames with spaces will work.

Save Settings: Saves the current Viewer settings to disk.



Chapter Fifteen

The OpusFTP Module

Because of the modular design of Opus, it is readily possible to incorporate new modules and custom handlers. With Opus 5.5 we have done this with a new FTP module which is designed to make access to a remote Internet FTP sites as easy as using your own hard drive.

To use Opus 5's FTP you must have a TCP/IP stack running using either AmiTCP, Interworks INET225 or similar. Of course if you want to access the Internet you must also be connected to a TCP/IP network (such as the Internet) via modem etc.

The OpusFTP module displays the remote Internet site using a standard Opus file Lister. Almost all the normal Opus actions and commands will function just as if this Lister was a local directory. You can copy to and from the remote site, delete files (if you have permission), rename files, double-click on a file to view or read it, and so on. You can also open and access as many remote sites at the one time as you wish. About the only thing you cannot do is copy files directly between two FTP listers because of complications with the FTP protocols.

If you are familiar with the Internet and FTP already you can now set up your address book using the configuration file.

Now all you have to do is to load the *FTP buttons* if they are not already there and click on *AddressBook* or *Connect*.

The FTP Configuration File

The file *DOpus5:System/ftp.config* will be read the first time you use Opus 5's FTP capabilities. This file contains two parts - the actual configuration options for OpusFTP, and the list of entries for the address book.

All lines in the configuration file beginning with a '#' character are considered comments and will be ignored.

Configuration Section

The template for the configuration options is:

**LIBRARY=LIB/K,LOG/K,LOGOFF/S,DEBUG/K/N,
TIMEOUT/K/N, LISTUPDATE/K/N**

You may put the options on separate lines to increase readability. All settings are optional, so OpusFTP will work even if you have no configuration file.

LIBRARY=LIB/K

This specifies which TCP/IP stack you wish to use. Currently supported keywords are:-

AMITCP	the AmiTCP/IP product from NSDI.
AS225	the old unreleased Commodore product.
INET	the I-Net 225 TCP/IP from Interworks.

AmiTCP is the default, but all will be checked so if you change which product you use, OpusFTP will still operate.

LOG/K

If you wish to log the actions of your FTP sessions, you may provide a filename here. More useful, however, is to supply a shell description so you can watch what is happening. Here are some sample Shell descriptions:-

For the most basic log window on the Workbench screen:-

LOG CON:

If you have the KingCon Shell by David Larsson (available on Aminet), the following will give you a scrollbar buffer and open on the Opus 5 public screen:-

LOG KCON:/400//100/FTP/AutoIconify/ScreenDOPUS.1

The default LOG entry is **CON:////Directory Opus 5 FTP**

The log file or shell specification can be changed at any time with the **FTPSetVar LogFile** command.

LOGOFF/S

Normally, if you specify a log file or shell, logging will be enabled by default. However, when you specify the '**LOGOFF**' keyword, logging will be turned off when you start using FTP. The log may be turned on and off at any time with the **FTPSetVar LogFile** command.

DEBUG/K/N

The debug option takes a value which specifies the amount of debugging information that will be output to the log. At present only zero and non-zero debugging levels exist. Zero means you only wish to see the responses from the FTP site you are connected to, and non-zero means you wish to see the commands sent by Opus 5 to the FTP site as well as the numeric response codes. *Debugging is set to zero by default.*

The debug level may be changed at any time with the **FTPSetVar Debug** command.

TIMEOUT/K/N

The TIMEOUT keyword allows you to specify how much time Opus 5 should wait for the FTP site to reply to each command.

The default timeout value is 60 seconds.

The timeout can be changed at any time with the ***FTPSetVar Timeout*** command.

LISTUPDATE/K/N

When you change directories on an FTP site, the lister is updated periodically as the filenames are received. This value specifies how often to update the lister. *The default list update value is 1 second.*

The list update period can be changed at any time with the ***FTPSetVar ListUpdate*** command.

Address Book Section

Each entry in your address book is specified on a separate line. The format of the line is:-

**ANONYMOUS=ANON/S, USERACCOUNT=ACCT/S,
ALIAS/K, HOST, ADDRESS=ADDR/K, PATH=DIR,
USER/K, PASSWORD=PASS/K**

ANONYMOUS=ANON/S

This optional switch specifies that the entry is for an anonymous FTP site.

USERACCOUNT=ACCT/S

This optional switch specifies that the entry is for an FTP site on which you have an account.

ALIAS/K

You may specify a nickname for each entry in the address book with this keyword. The address book will display the nickname if it exists or the hostname or IP address if not.

HOST

This is the hostname, or actual Internet address of the machine you wish to connect to. For example, to access the main Aminet FTP site you could use **HOST=ftp.wustl.edu** or because the keyword is optional you could just use **ftp.wustl.edu**

ADDRESS=ADDR/K

This is the IP number of the machine you wish to connect to. It is often quicker to connect to an Internet site by IP number than it is by hostname. For example, to access the main Aminet FTP site you could use **ADDRESS=128.252.12.1**

PATH=DIR

This is the directory on the FTP site where you wish to begin. For example, most Aminet sites this is **/pub/aminet**

USER/K

If you wish to connect to a specific account on an FTP site, this is where you can specify your account name. For example, if your email address is 'fred@foo.bar.com' and you wish to make an entry for your account on the machine 'foo.bar.com' then you would use the name 'fred' here.

PASSWORD=PASS/K

Usually, when you use an FTP account other than 'anonymous' or 'ftp', you will need to supply a password. You can put your password here for fully automatic log-ins if you wish. For security reasons it is a much better idea not to leave any passwords in any configuration files on your machine. If you connect to a site using a username but no password, you will be prompted for your password at that time.

Opus 5 FTP Commands

Trapped Commands

OpusFTP transparently supports most of the standard built-in Opus commands. The following commands are supported directly with some minor differences:-

Copy	CopyAs
Delete	MakeDir
Parent	Rename
Root	

Copy and **CopyAs** do not copy entire directories, only files.

Copying an existing file where the new file is larger than the old provides a *resume* feature.

Delete usually only deletes directories if they are empty, but this does depend on the server's operating system.

Rename does not allow wildcards.

Some commands are supported by downloading the selected files into the T: directory then executing the command on the temporary files. These include:-

AnsiRead	DoubleClick
HexRead	Play
Print	Read
Run	Show
SmartRead	

Internal OpusFTP Commands

The following commands are added to the internal Opus command list by the OpusFTP module. You may use these in buttons, menus, scripts etc, just like standard Opus commands.

System Commands

FTPAddressBook

The *FTPAddressBook* command displays a requester which allows you to connect to any FTP site listed in your configuration file by double-clicking on the entry or selecting it and clicking the *OK* button.

FTPConnect HOST, USER, PASSWORD=PASS, DIR/K

The *FTPConnect* command provides requesters for entering the details of the FTP site you wish to connect to.

When prompted, enter the host name or the IP number of the remote site. Next, enter your user name. This will usually be 'anonymous' but you may enter your login name if you have an account on the remote machine.

If you have entered a user name other than 'anonymous' you will also be prompted for the password of your account.

FTPCommand [<arguments>]

This command lets you send an FTP command directly to the server of the current 'SRCE' FTP lister. This command is provided mainly for experts who wish to set site options. You can experiment with FTP commands like HELP, STAT, and PWD. A requester will be provided if the arguments are omitted.



This is an advanced command. It is possible to confuse Opus when using this command so be careful!

FTPSetVar <variable> [<value>]

The *FTPSetVar* command lets you set the value of the internal Opus 5 FTP variables.

LogFile [<name>]: If <name> is specified, a new log will be opened using the new name. If a log was already open, it will be closed. If <name> is not specified, the log will be toggled

between open/closed using the original name in the configuration file:-

FTPSetVar LogFile "CON:////Opus FTP Log"
FTPSetVar LogFile

Debug [<level>]: If <level> is provided, the debugging level will be set to this new value. If <level> is not provided, the debugging level will be toggled between 0 and 1. <level> must be an integer:-

FTPSetVar Debug 1
FTPSetVar Debug

Timeout <seconds>: <seconds> must be provided. This sets the maximum amount of time Opus will wait for replies from the server before giving up and disconnecting:-

FTPSetVar Timeout 60

ListUpdate <seconds>: <seconds> must be provided. This sets the interval in seconds between successive updates of the lister while a new directory is being scanned:-

FTPSetVar ListUpdate 5

FTPQuit FORCE/S

The *FTPQuit* command causes all connected listers to log out as soon as they can, closes the log if it is open, and allows the FTP module to be removed from memory when needed.

If the **FORCE** keyword is supplied, all connected Listers will abort their current operations and log out immediately.

Transfer Commands

Use the standard Opus "trapped" commands instead of these if possible.

FTPCD [<dirname>]

The *FTPCD* command allows you to change the current SRCE FTP lister to a different directory. If no directory name is specified, a prompt requester will be provided.

FTPCopy [<from>] [<to>]

The *FTPCopy* command allows you to copy a file from the current SRCE FTP lister to the current DEST non-FTP lister. Prompt requesters will be provided if the filenames are omitted.

FTPRename [<oldname>] [<newname>]

The *FTPRename* command allows you to rename an entry on the current SRCE FTP lister. Requesters will be provided if the filenames are omitted.

FTPDelete [<name>]

The *FTPDelete* command allows you to delete an entry from the current SRCE FTP lister. A requester will be provided if the filename is omitted.

Resume

The Opus 5 FTP module includes a handy *Resume* feature which can be useful if you have a slow or unreliable connection, or have to disconnect in the middle of a transfer, or even if you suffer a crash!

All you need to do is re-establish the connection at some later time and locate the same file - it doesn't have to be in the same place as long as it is exactly the same as the original. Then simply select the file you wish to resume, make sure the original partial file is in the

destination lister and click the *Copy* button. A requester will appear giving you several options. You may skip or replace the whole file as usual in Opus, or click on *Resume* to receive (or send) the rest of the file.

This requester automatically appears whenever you attempt to copy a file to or from a normal Opus directory when a file with an identical name already exists and the source file is larger than the destination file.



Unfortunately, not all FTP sites support the Resume operation. This depends on the remote operating system. Some like WindowsNT do, but they Resume from byte 0!



Chapter Sixteen

ARexx

The Directory Opus 5 ARexx port name is **DOPUS.x**, where x is the invocation count of the program (the first and most often used one is **DOPUS.1**). Since ARexx scripts launched from Directory Opus do not automatically inherit the command address, you may want to use the **{Op}** command sequence in Opus functions (this is described elsewhere in the manual).

If a command returns a value or information, the data will generally be returned in the **RESULT** variable. The only exceptions to this are the **dopus getstring** and **lister getstring** commands (see below) which return information in the special **DOPUSRC** variable. Error codes are returned in the **RC** variable.

The ARexx command set is very comprehensive and flexible. Almost complete control of listers in name mode is offered, along with the ability to launch functions and commands. You can add your own commands to Opus via ARexx scripts which are loaded automatically - the commands appear identical to the built-in functions. You can even replace the default commands with your own.

There is also a powerful *custom handler* ability. This allows your ARexx program to receive messages from Opus for a variety of user actions, including lister and icon events. See the section on *Custom Handlers* on page 241 for more information.

ARexx Commands

For simplicity, the Directory Opus 5 command set is arranged in a hierarchical structure, with only three main (or base) commands:- **dopus**, **lister** and **command**.

dopus

The first base command is **dopus**. This is a general purpose command, and allows you to perform functions not falling into the other categories.

- **dopus addappicon <port> <label> <id> [pos <x> <y>] [icon<filename>] [quotes] [info] [snap] [close] [local] [locked] [menu <stem>] [base <base>]**

This command allows you to add your own AppIcons to the DOpus (and optionally the Workbench) screen from ARexx. You can specify an icon image to use, the label of the icon and the position on the screen, as well as several other parameters. You can also specify the items for the pop-up menu for the new icon. Opus sends messages to the message port you specify - see the example scripts provided for more information on how to receive and process these messages.

The parameters for "addappicon" are:-

- | | |
|---------------|---|
| port | - the name of the port messages are sent to |
| label | - the icon label (text displayed under the icon) |
| id | - your own ID for the icon; this is returned in messages |
| pos | - position for the icon (xy coordinates) |
| icon | - optional pathname of icon file to use (without the .info suffix) |
| quotes | - specify this keyword if you want filenames quoted when sent to the message port |
| info | - if you want <i>Information</i> to work on this icon |
| snap | - if you want <i>Snapshot</i> to work on this icon |
| close | - if you want <i>Close</i> instead of <i>Open</i> in the pop-up menu |

- local** - if you want the icon to be local to Opus (and not appear on Workbench)
- locked** - the icon will start out *locked* in position, unable to be moved
- menu** - name of a stem variable containing your own menu items
- base** - allows you to specify the base ID of messages sent from the pop-up menu

The **menu** parameter allows you to specify your own items for the icon's pop-up menu. The stem variable must be in the following format:

stem.COUNT - number of items
stem.0 - item 1
stem.1 - item 2
etc.

If you specify "---" as an item, a separator bar will appear. When you receive a message that the user has selected one of these menu items, the message will contain the ID of the item. This is the value corresponding to the item's position in the stem array (eg 0 for item 1, 1 for item 2, etc). If the *base* field is specified, the value given for the base will be added to this ID.

This command returns an appicon handle in **RESULT** if it is successful. This same handle can be passed to *dopus setappicon* to modify the icon, and must be passed to *dopus remappicon* to remove the appicon when you are finished. If you do not specify an icon file to use then the system's default "tool" icon will be used for the image.

Messages from the appicon are sent to the port you specified. The messages are structured in much the same way as messages sent from listers, so there's no reason you shouldn't be able to use the same code for both. For more information, see the *Custom Handlers* section on page 241.

- **dopus addtrap** [**abort** | **<command>**] **<handler>** [**port** **<portname>**]

This command allows your script to trap the progress bar's abort button or any Opus internal command. Specify the *abort* keyword to trap the abort message, or the name of the internal command you wish to trap. *handler* is the name of your custom handler message port. If you pass the name of a message port with the optional *port* keyword, the message will be sent to that port instead of your usual handler port. This can be very useful when used with *abort* if your handler is busy doing something synchronously when the abort is pressed. If the abort port is on a separate process, it may be able to interrupt the main handler process using signals for instance. The *dopus remtrap* command is used to remove trapped functions. See the *Custom Handlers* section on page 241 for more information on the messages sent.

- **dopus back**

This command moves the Directory Opus 5 window (and screen) to the rear of the display.

- **dopus command** **<name>** **program** **<scriptname>** [**desc** **<description>**] [**template** **<template>**] [**source**] [**dest**]

This command provides the ability to add new internal commands to Directory Opus, or to replace existing commands. It is generally called from within the *init* function of an Opus ARexx module and the *program* parameter will be the name of that module. The *program* field is mandatory, and Opus will run the script name you provide here whenever this function is invoked. See *ARexx modules* and the example script on page 248 for more information.

- **dopus error** **<code>**

This command is used to retrieve meaningful error messages when passed an Opus ARexx error code. For example,

- > dopus error 1
--> File rejected by filters
- > dopus error 10
--> Invalid lister handle

- **dopus front**

This command moves the Directory Opus 5 window (and screen) to the front of the display.

- **dopus getfiletype <filename> [id]**

This command allows you to query a file to see if it is recognised by Directory Opus 5. <filename> is the name of the file, including the full path. By default, if the file is recognised the Filetype description string will be returned in **RESULT**. If you specify the *id* keyword, the Filetype ID will be returned instead. For example,

- > dopus getfiletype ram:testfile.lha
--> LHA Archive
- > dopus getfiletype ram:picture.jpg id
--> JPEG

- **dopus getstring <text> [secure] [<length>] [<default>] [<buttons>]**

This command allows you to prompt the user to input a text string. <text> is a string of text to be displayed in the requester, and should be surrounded by quotes if it contains spaces. The *secure* keyword causes the string to be displayed as asterisks, which can be useful for passwords. <length> is the maximum length of the string to accept and defaults to 80 if not specified. <default> is the default value of the string; that is, the text you wish to initially appear in the field. <buttons> are the buttons you wish the requester to have; each button should be separated by a vertical bar character. If the buttons parameter is omitted the requester will have a single button marked 'OK'. For example,

- > dopus getstring "Please enter some text" 40 "" "Okay | Cancel"

This would display a requester with the string "*Please enter some text*", a maximum input length of 40 characters, no default text, and buttons labelled *Okay* and *Cancel*.

The string (if any) is returned in **RESULT**. The cardinal number of the selected button is returned in the special variable **DOPUSRC**. In the above example, if the user clicked *Okay*, **DOPUSRC** would contain 1, and if the user clicked *Cancel* it would contain 0. This command and the *lister getstring* command are the only ones that use the **DOPUSRC** variable currently, but this may change in the future.



Please note that previous versions of Opus 5 did not clear RESULT if an empty string was entered. Make sure that this change does not affect your scripts.

- **dopus read <filename>**
- **dopus read <handle> <filename>**
- **dopus read <handle> quit**

The read command is provided to allow you greater control over the Opus text viewer. They basically allow you to open a viewer and view multiple files in it one by one before closing it again. You use the viewer handle in much the same way as you would a lister handle. For example,

```
> dopus read ram:file1.txt
--> 121839492
> dopus read 121839492 ram:file2.txt
--> 121839492
> dopus read 121839492 quit
--> 0
```

- **dopus remappicon <handle>**

Removes an appicon that was added previously with the *dopus addappicon* command.

- **dopus remtrap** [**abort** | **<command>**] **<handler>**

Disables the trapping of the progress bar's abort button or the specified Opus internal command as initiated with the *dopus addtrap* command. If you specify "*" as the command, all traps added for this handler will be removed. *handler* is the name of the message port as specified in *the dopus addtrap* command.

- **dopus request** **<text>** **<buttons>**

This command allows you to request a choice from the user. **<text>** is a string of text to be displayed in the requester. **<buttons>** are the buttons you wish the requester to have; each button should be separated by a vertical bar character. For example,

```
> dopus request "Please choose an option" "  
Option 1 | Option 2 | Option 3" (all one line)
```

This would display a requester with the string "Please choose an option", and three buttons labelled Option 1, Option 2 and Option 3.

The cardinal number of the selected button is returned in **RC**. The last button supplied (Option 3 in this case) is designated a *Cancel* button, and so returns the value 0. Therefore, the values returned by this example are 1, 2 and 0 respectively.

- **dopus screen**

This command returns information about the Opus screen in the following format:-

```
<name> <width> <height> <depth> <barheight>
```

If Opus is iconified it does not have a screen. In this case **RC** will be set to 5. You can use this to find out whether or not Opus is currently iconified. Here is an example of the output:

```
> dopus screen  
--> DOPUS.1 640 512 2 10
```

The *barheight* value is useful if you intend to open listers or other windows just below the screen title bar.

- **dopus send** <port name> <string>

This command does nothing to DOpus itself, but instead makes it easy for you to send a string (or any length) to another ARexx task (via a message). The string is supplied in Arg0 of the message sent to the named port.

- **dopus setappicon** <handle> <item>

This allows you to do things to AppIcons added with the *dopus addappicon* command. Valid <item> fields are :-

text <text>

Change the icon label.

busy [on | off]

Make icon busy or non-busy.

locked [on | off]

Make icon locked or unlocked.

If an icon is *busy*, it is unselectable by the user. It will not respond to double-clicks, pop-up menu events, drag'n'drops, etc. The icon image is ghosted when it is busy. When an icon is *locked*, its position can not be changed and it can not be moved manually by the user, nor will a CleanUp will not affect it.

- **dopus version**

The version command returns a string in the format <version> <revision> and is useful in ARexx scripts for determining if certain features exist.

lister

The next base command, `lister`, allows you to control listers and entries within listers.

- **`lister add <handle> <name> <size> <type><seconds> <protect> <comment>`**

This command adds an entry to the specified lister. `<name>` is the full name of the entry; `<size>` is the size of the entry; `<type>` is the type of the entry (less than 0 for a file, greater than 0 for a directory); `<seconds>` is the datestamp of the entry in seconds from 1/1/78; `<protect>` is the protection bits of the file (in ascii format); `<comment>` is the comment of the entry (if any). Valid entry types are:-

- 0 device
- 1 plain directory
- 1 plain file
- 2 directory in *assign* colour
- 2 file in *device* colour
- 3 directory in bold (link)
- 3 file in bold (link)
- 4 directory in *assign* colour and bold
- 4 file in *device* colour and bold



After a 'lister add' command, the display is not updated until you execute a 'lister refresh' command.

For example,

```
> lister add 121132636 "My file!" 12839 -1 540093905  
prwdd my comment
```

- **`lister addstem <handle> <stem>`**

This command adds files to a lister via a stem variable. It is more powerful than the `lister add` command and should be used in preference. The fields of the stem variable are very similar to those returned by a `lister query <handle> entry stem` command (in fact, you could pass the result the query

directly to an *addstem* to add an identical entry to another lister.) The fields that are used are :-

name	- name of entry
size	- file size
type	- type of entry (same as for <i>lister add</i>)
protstring	- protection bits (ASCII string, eg "rwed")
protect	- protection value (number, used if <i>protstring</i> is not given)
comment	- file comment
datestring	- creation date and time (ASCII string)
date	- number of seconds since 1/1/78 (used if <i>datestring</i> is not given)
filetype	- ascii string for file type display
selected	- 0 or 1
version	- version number
revision	- revision number
verdate	- version date string
userdata	- user data (value, not a string)
display	- custom display string
menu	- custom pop-up menu
base	- base ID for pop-up menu

Not all of these fields are required. As a bare minimum you should specify either the *name* or the *display* field.

The *display* string allows you to specify a completely custom string to display for the entry. None of the other information will be displayed if this string is supplied. The maximum length is 256 characters.

The *userdata* field allows you to specify your own ID value (or any other value) to be associated with this entry. Its main usage is with the custom pop-up menu and custom handlers.

The *menu* field allows you to specify a stem variable containing custom items for the pop-up menu that appears when the user presses the right button on this entry. Its format is the same as for AppIcon pop-up menus;

stem.COUNT - number of entries
stem.BASE - base ID

stem.0 - entry 1
stem.1 - entry 2

If *count* is set to 0, right-button pop-ups will be disabled for this file. If this field is not specified, the default pop-up menu will be displayed. If you specify "---" as an item, a separator bar will appear. When you receive a message that the user has selected one of these menu items, the message will contain the ID of the item. This is the value corresponding to the item's position in the stem array (eg 0 for item 1, 1 for item 2, etc). If the *base* field is specified, the value given for the base will be added to this ID.

See the *Custom Handlers* section later in this chapter for more information on the messages sent.

- **lister copy <handle> <destination>**

This command copies the contents of one lister to another lister. Unlike most commands, the display of the destination lister is refreshed immediately. For example,

```
> lister copy 121132636 121963868
```

- **lister clear <handle>**

This command clears the contents of the specified lister. The display will not be updated until you execute a *lister refresh* command.



In previous versions of Opus 5, this command also cleared the custom handler name. This is no longer the case.

- **lister clear <handle> <item> <value>**

This command clears a particular item of information in the specified lister. <handle> is the handle of the lister in question; <item> can be one of the following keywords:-

abort

This clears the abort flag in the specified lister.

> lister clear 121132636 abort

flags <flags>

Clears sort/display flags for this lister. The display is not updated unless you execute a lister refresh command. See the *lister query* section for the keywords to use. For example,

> lister clear 121132636 flags reverse

progress

This turns the progress indicator off in the specified lister.

> lister clear 121132636 progress

- **lister close [all | <handle>]**

This command closes the specified lister or all listers if the *all* keyword is supplied in place of a lister handle. Any function that is currently taking place will be aborted. <handle> is the lister handle that was returned when you created this lister with the *lister new* command. For example,

> lister close 121132636

- **lister empty <handle>**

This command will display an empty cache in the specified lister (unlike *lister clear* which clears the contents of the current cache). If no empty caches are available (and a new one can not be created), the existing cache will be cleared. If the lister has a custom handler attached, it will receive an *inactive* message.



Note that previous versions of Opus 5 did not send the 'inactive' message to the custom handler when this command was used.

- **lister getstring** <handle> <text> [secure] [<length>]
[<default>] [<buttons>]

This command is identical to the *dopus getstring* command except that it takes a lister handle as an additional parameter and the requester will be centred over that lister.

<text> is a string of text to be displayed in the requester, and should be surrounded by quotes if it contains spaces. The *secure* keyword causes the string to be displayed as asterisks, which can be useful for passwords. <length> is the maximum length of the string to accept and defaults to 80 if not specified. <default> is the default value of the string; that is, the text you wish to initially appear in the field. <buttons> are the buttons you wish the requester to have; each button should be separated by a vertical bar character. If the buttons parameter is omitted the requester will have a single button marked 'OK'. For example,

```
> lister getstring 121132636 "Please enter some text"  
40 "" Okay | Cancel'
```

This would display a requester over the lister with the string "Please enter some text", a maximum input length of 40 characters, no default text, and buttons labelled Okay and Cancel.

The string (if any) is returned in **RESULT**. The cardinal number of the selected button is returned in the special variable **DOPUSRC**. In the above example, if the user clicked Okay, **DOPUSRC** would contain 1, and if the user clicked Cancel it would contain 0. This command and the *dopus getstring* command are the only ones that use the **DOPUSRC** variable currently, but this may change in the future.

- **lister new** [<x/y/w/h>] [toolbar <toolbar>] [<path>]

This command creates a new lister. You may optionally specify the position and size of the new lister. The default position is -1/-1 causes the lister to open under the mouse pointer. A custom toolbar for the lister can be specified with the *toolbar* keyword; toolbar files are expected to be found in

the *DOpus5:Buttons* directory if the full path is not given. You may also specify a path to read when the lister opens. For example,

```
> lister new
> lister new 100/50/400/300
> lister new ram:
> lister new 80/30/200/200 dh0:work
> lister new toolbar custom_toolbar work:
--> 121132636
```

If the lister opens successfully, its **handle** is returned in the **RESULT** variable. You must save the value of this handle if you wish to do anything further with this lister. In the above example, a handle of 121132636 was returned. This will be used for further examples below.

- **lister query <handle> <item>**

This command returns a particular item of information from the specified lister. *<handle>* is the handle of the lister in question. All information is returned in the **RESULT** variable, unless an error occurs. *<item>* can be one of the following keywords:-

all

This command returns the handles of all non-busy listers (that is, any listers that are not performing a function at the time). Note that this does not require a lister handle to operate. This also supports the use of stem variables. For example,

```
> lister query all
--> 121132636 121963868
```

abort

This returns a boolean value indicating the status of the lister's abort flag. This query command is only valid if the lister has a progress indicator open (as this is the only way the user can abort a function anyway). This will return 1 if

the user has clicked the abort gadget, 0 if she has not. For example,

```
> lister query 121132636 abort
--> 0
```



Note that in Opus 4, querying the abort flag would also reset it. This is not the case in Opus 5; if you wish to reset the state of the abort flag you must use the "lister clear" command.

busy

Returns a boolean value (0 or 1) indicating the lister busy status. That is, if the lister is currently busy, it will return 1, otherwise it will return 0. For example,

```
> lister query 121132636 busy
--> 1
```

dest

This command returns the handles of all destination listers currently open. Note that this does not require a lister handle to operate. This also supports the use of stem variables. For example,

```
> lister query dest
--> 121963868
```

This command supports the **stem** and **var** keywords (see *lister query entry* for more information.)

dirs <separator>

Returns the names of all directories in the lister. For example,

```
> lister query 121132636 dirs
--> "Clipboards","ENV","T"
```

This command supports the **stem** and **var** keywords (see *lister query entry* for more information.)

display

This returns a string indicating the current display items. The string will consist of the same keywords as for sort, in the order that they appear in the lister (if they appear at all). For example,

```
> lister query 121132636 display
--> name size date protect comment
```

entries <separator>

Returns the names of all entries (that is, both files and directories) in the lister. For example,

```
> lister query 121132636 entries
--> "Clipboards" "ENV" "T" "abc" "Disk.info"
```

This command supports the **stem** and **var** keywords (see *lister query* entry for more information.) For example,

```
> lister query 121132636 entries stem files
```

This would return the following variables :-

```
files.count = 7
files.0 = Clipboards
files.1 = ENV
files.2 = T
files.3 = abc
etc.
```

entry <name>

Returns information about the specified entry. *<name>* is the actual name of the entry to return information about. You can supply #xxx for the name (where xxx is a number), to specify the cardinal number of the desired entry. This command can return information in two ways. The default way is to return a string of information in either the **RESULT** variable or another variable of your choice. The information returned in this case is

<name> <size> <type> <selection> <seconds>
<protect> <comment>

where <name> is the full name of the entry, <size> is the size of the entry, <type> is the type of the entry (<0 means a file, >0 means a directory), <selection> indicates the selection status of the entry (1 if the entry is selected, 0 if it is not selected), <seconds> is the timestamp of the entry in seconds from 1/1/78, <protect> is the protection bits of the file (in ascii format); and <comment> is the comment of the entry (if any). For example,

```
> lister query 121132636 entry ENV
--> ENV -1 2 0 543401724 ----rwed
```

By default the string is returned in the **RESULT** variable. If you wish to use another variable name, specify the **var** keyword followed by the variable name. For example,

```
> lister query 121132636 entry ENV var my variable
```

The second, and more elegant method, returns information about the entry in a **stem variable**. To use this second method, you must specify the **stem** keyword followed by the name of the stem variable you wish to use. For example,

```
> lister query 121132636 entry ENV stem fileinfo
```

The specified stem variable will have several fields, each containing information about the entry in question. These fields are as follows:-

name	- filename
size	- file size
type	- type (<0 = file, >0 = dir)
selected	- 0 or 1
date	- seconds since 1/1/78
protect	- protection bits (long value)
datestring	- timestamp in ascii form
protstring	- protection bits in ascii form
comment	- file comment (if any)

filetype	- file type (if any)
version	- version number
revision	- revision number
verdate	- version date (numerical dd.mm.yy format)
datenum	- file date in numerical dd.mm.yy format
time	- file time in hh:mm:ss 24 hour format

Several other *query* commands in this section support the **var** and **stem** keywords.



*Note regarding the **lister query files**, **dirs**, **entries**, **selfiles**, **seldirs**, and **selentries** commands. In previous versions of Opus 5, **RESULT** was not cleared if there were no entries to return. This problem has been fixed. Also when using these commands with the **STEM** keyword, the **COUNT** field will now be set to zero in this case.*

files <separator>

Returns the names of all files in the lister. The names are returned as one long string, separated by spaces. You may change the separation character by specifying it after the **files** keyword. For example,

```
> lister query 121132636 files
--> "abc" "Disk.info" "readme" "zzz.zzz"
```

This command supports the **stem** and **var** keywords (see *lister query entry* for more information.)

firstsel

Returns the name of the first selected entry in the lister. The entry is not deselected, so if you don't deselect it yourself this command will only ever return the one name. For example,

```
> lister query 121132636 firstsel
--> "ENV"
```

flags

This returns a string indicating any sort or display flags that are active for the lister. These flags are:-

reverse	- sort in reverse order
noicons	- filter icons
hidden	- filter hidden bit

For example,

```
> lister query 121132636 flags
--> noicons
```

hide

This returns the current hide filter for this lister. For example,

```
> lister query 121132636 hide
--> #?.o
```

handler

Returns the name of the current custom handler port. For example,

```
> lister query 121132636 handler
--> ArcDir121132636
```

label

This command returns the label that appears beneath this lister when it is iconified. By default, the label will be the name of the current directory. This label can however be changed by calling the *lister set label* command. For example,

```
> lister query 121132636 label
--> Ram Disk
```

lock <type>

This command returns to current lock status of the lister where <type> is either *state* or *format*. See the *lister set lock* command.

mode

This returns the current mode for this lister and also the word *showall* if the lister is in an icon mode and displaying files without icons. The lister modes are:-

name - name mode
icon - workbench style icon mode
icon action - icon action mode

For example,

```
> lister query 121132636 mode
--> icon action showall
```

numdirs

Returns the number of directories in the lister. For example,

```
> lister query 121132636 numdirs
--> 3
```

numentries

Returns the total number of entries in the lister (files + dirs). For example,

```
> lister query 121132636 numentries
--> 7
```

numfiles

Returns the number of files in the lister. For example,

```
> lister query 121132636 numfiles
--> 4
```


numselentries

Returns the number of selected entries in the lister.

numseldirs

Returns the number of selected directories in the lister.

numselfiles

Returns the number of selected files in the lister.

path

Returns a string indicating the current path visible in the lister. For example,

```
> lister query 121132636 path
--> ram:
```

position

Returns the current position and size of the lister. The word *locked* will also be returned if the lister is locked in position. For example,

```
> lister query 121132636 position
--> 80/30/200/200 locked
```

selfiles <separator>

Returns the names of all selected files in the lister. This command supports the **stem** and **var** keywords (see *lister query entry* for more information.)

seldirs <separator>

Returns the names of all selected directories in the lister. This command supports the **stem** and **var** keywords (see *lister query entry* for more information.)

selentries <separator>

Returns the names of all selected entries (ie both files and directories) in the lister. This command supports the **stem** and **var** keywords (see *lister query entry* for more information.)

separate

This returns a keyword indicating the current file separation method in this lister. Valid methods are:-

mix	- mix files and directories
dirstfirst	- directories first
filesfirst	- files first

For example,

```
> lister query 121132636 separate
--> dirsfirst
```

show

This returns the current show filter for this lister.

source

This command returns the handles of all source listers currently open. Note that this does not require a lister handle to operate. For example,

```
> lister query source
--> 121132636 128765412
```

This command supports the **stem** and **var** keywords (see *lister query entry* for more information.) For example,

```
> lister query source stem sources.
```

This would return:-

```
sources.count = 2
```

```
sources.0 = 121132636
sources.1 = 128765412
```

sort

This returns a keyword indicating the current sort method in this lister. Valid sort methods are:-

name	- filename
size	- file size
protect	- protection bits
date	- datestamp
comment	- comment
filetype	- file type
version	- file version

For example,

```
> lister query 121132636 sort
--> name
```

toolbar

This returns the toolbar currently being used by this lister. For example,

```
> lister query 121132636 toolbar
--> DOpus5:Buttons/toolbar
```

visible

Returns a boolean value indicating if the lister is currently visible. For example,

```
> lister query 121132636 visible
--> 1
```

- **lister read <handle> <path> [force]**

This command will read the given path into the specified lister. By default a new cache is used to read the directory; if the *force* keyword is specified, the current cache will be

cleared and the directory will be read into that. The old path is returned in **RESULT**. For example,

```
> lister read 121132636 'dh0:test'  
--> RamDisk:
```

- **lister refresh (all | <handle>) [full] [date]**

This command refreshes the display of the specified lister or all listers if the keyword *all* is given in place of a lister handle. Unlike Opus 4, none of the lister modifying commands above will actually refresh or update the lister display; hence, you must use this command after making any changes (changing sort method, adding files, etc) to have the changes display. The optional *full* keyword causes the lister title and status displayed to be refreshed as well. For example,

```
> lister refresh 121132636 full
```

If the *date* keyword is specified, the lister will update its directory timestamp, which will stop it re-reading itself the next time it is activated. If this keyword is specified, the lister display itself is not refreshed.

```
>lister refresh 12113236 date
```

- **lister remove <handle> <name>**

This command removes an entry from the specified lister. <name> is either the name of the entry, or #xxx (where xxx is a number) to specify the cardinal number of the entry. The display is not updated until you execute a lister refresh command. For example,

```
> lister remove 121132636 #5
```

- **lister request <handle> <text> <buttons>**

This command is identical to the *dopus request* command except that it takes an additional *handle* parameter and the requester will be centred over that lister.

`<text>` is a string of text to be displayed in the requester.
`<buttons>` are the buttons you wish the requester to have; each button should be separated by a vertical bar character. For example,

```
> lister request 121132636 "Please choose an option" "  
Option 1 | Option 2 | Option 3"
```

This would display a requester with the string "Please choose an option", and three buttons labelled Option 1 to Option 3.

The cardinal number of the selected button is returned in RC. The last button supplied (Option 3 in this case) is designated a *Cancel* button, and so returns the value 0. Therefore, the values returned by this example are 1, 2 and 0 respectively.

- **lister set `<handle>` `<item>` `<value>`**

This command sets a particular item of information in the specified lister. `<handle>` is the handle of the lister in question. `<item>` can be one of the following keywords:-

busy `<state>` [`wait`]

Sets the busy status for this lister. You can specify 0 or *off* to turn the busy pointer off, or 1 or *on* to turn it on. When turning busy status on you can also provide the *wait* keyword which will cause the command to be synchronous. For example,

```
> lister set 121132636 busy on wait  
> lister set 121132636 busy 0
```

case

This command turns on or off case sensitivity for this lister. Since Amiga filenames are not case sensitive this setting defaults to off. It may be useful for some custom handlers to turn case sensitivity on however.

```
> lister set 121132636 case on  
> lister set 121132636 case off
```

dest [lock]

Makes this lister the destination. If you specify the lock keyword, it will be locked as a destination. For example,

```
> lister set 121132636 dest
```

display <items>

Sets the display items for this lister. The display will not be updated until you execute a *lister refresh* command. See the *lister query* section for the item keywords to use. For example,

```
> lister set 121132636 display name date size protect
```

field [<number> <string>]

This allows you to set your own strings to be used in the lister field titles. You can not change the nature of the columns in the lister - this just allows you to change the heading.

The <number> (0-9) specifies which string to replace; counting from 0 they are:-

name, size, access, date, comment, type,
owner, group, net, version

Set to an empty string to restore the default. You will need to do a *lister refresh <handle> full* to update the display once you have changed the titles.

You can also do *lister set <handle> field off* to turn field titles off altogether, and *on* to turn them back on again. Note that if field titles have not been enabled in the configuration, an ARexx script is unable to turn them on.

For example,

```
> lister set 121132636 field 0 FileName 4 Notes
```

flags <flags>

Sets sort/display flags for this lister. The display is not updated unless you execute a lister refresh command. See the *lister query* section for the keywords to use. For example,

```
> lister set 121132636 flags reverse noicons
```

header <string>

This works just like *lister set title* except it changes the text in the 'Files x/y Dirs x/y' bar. The old header string is returned in RESULT. Set this to an empty string to restore the default. If you wish to actually display an empty header, set it to a - (hyphen character.)

handler <port name> [quotes] [fullpath]

Sets the custom handler port name for this lister. This is the name of the message port to which messages from Opus will be sent. If you specify the *quotes* flag, any filenames sent in messages to the port will be enclosed in quotes (this is a good idea as it allows you to support filenames containing spaces). If you specify the *fullpath* flag, messages will always contain the full path name of a file, irrespective of whether it came from an Opus lister or not. (Usually, if the file comes from a lister you will only get the filename itself, plus the lister handle with which to find out the path). For example,

```
> lister set 121132636 handler 'lhadir_handler' quotes
```

See *Custom Handlers* later in this chapter for more details.

hide <pattern>

Sets the hide pattern for this lister. The pattern is applied immediately but the display is not updated until you execute a lister refresh command. For example,

```
> lister set 121132636 hide '?.info'
```

label

This command can be used to set the label that will be displayed beneath the iconified lister. To remove a custom label, simply use this command with no label specified. For example,

```
> lister set 121132636 label Custom Lister  
> lister set 121132636 label
```

lock <type>

The type parameter may currently be:-

state [on | off]

The *state* parameter allows you to lock the lister to its current state so the user will be unable to change it until you unlock it.

format [on | off]

The *format* parameter allows you to lock the lister to its current display format. Currently this just prevents the user bringing up the Format Edit Requester.

You can string these commands on the one line, for example,

```
> lister set 121132636 lock state on format on
```

mode

This command sets the mode for this lister. See the *lister query* section for the keywords to use. For example,

```
> lister set 121132636 mode name  
> lister set 121132636 mode icon action showall
```


namelength

This command sets the maximum length allowed for filenames in this lister. The minimum length is 30 characters which is also the default length. This command will only be useful for writers of custom handlers. Note that the internal Opus commands, for the most part, do not currently support filenames longer than 30 characters. For example,

```
> lister set 121132636 namelength 256
```

newprogress [name] [file] [info] [bar] [abort]

This turns the progress indicator on in the specified lister. This is similar to the old *lister set progress* command, but allows greater control over the information displayed.

name	- allocates space for filename display
file	- allocates space for file progress display
info	- allocates space for information line
bar	- allocates space for progress bar
abort	- adds abort gadget

Progress windows that show both the bar graph and the file progress will have the graph and file displays swapped around. This means that instead of the graph showing the percentage of files copies, and a 'xx%' display showing the progress of that file, the graph shows the file progress and a 'xxx of yyy' display gives overall information.

```
> lister set 121132636 newprogress name file info bar abort
```

newprogress name <filename>

If the progress bar was opened with the *name* parameter, this will set the current filename.

```
> lister set 121132636 newprogress name 'myfile.txt'
```

newprogress file <total> <count>

If the progress indicator was opened with the ***file but not the bar*** parameter, this will set the total number of files and the number of the current file. This is shown as 'xx%' in the top right of the requester.

If the progress indicator was opened with ***both the file and bar*** parameters, this will set the total number of bytes and the current byte count. This is shown in the bar graph part of the requester.

```
> lister set 121132636 newprogress file 12 4
```

newprogress info <text>

<text> is a text string to be displayed between the filename and the bar graph of the progress indicator. For example,

```
> lister set 121132636 newprogress info "From 'T' to 'Ram:'"
```

newprogress bar <total> <count>

If the progress indicator was opened with the ***bar but not the file*** parameter, this will set the total number of bytes and the current byte count. This is shown in the bar graph part of the requester.

If the progress indicator was opened with ***both the file and bar*** parameters, this will set the total number of files and the number of the current file. This is shown as 'xxx of yyy' in the top right of the requester.

```
> lister set 121132636 newprogress bar 1024 100
```

newprogress title <text>

<text> is a text string to be displayed in the title bar of the progress indicator. For example,

```
> lister set 121132636 newprogress title 'Copying...'
```



You can use the old 'lister set progress' commands on a 'newprogress' indicator, but obviously they can only change the filename and bar count. Use 'lister clear progress' to remove either the 'old' or 'new' progress indicators.

off

Turns this lister off (ie neither source nor destination). For example,

```
> lister set 121132636 off
```

path <path string>

Sets the current path string in the lister. Note that this does NOT cause the directory to be read, it merely changes the displayed string. To read a new directory, use the lister read command. For example,

```
> lister set 121132636 path 'dh0:work'
```

position <x/y/w/h>

This sets the current position and size of the lister if it has not been locked. If the lister is visible the window will be moved immediately. For example,

```
> lister set 121132636 position 20/20/400/300
```

progress <total> <text>

Please note that the 'lister set progress' commands have now been superseded by the 'lister set newprogress' commands. Please use those commands in any new scripts.

This turns the progress indicator on in the specified lister. <total> specifies the total amount to be processed, and controls the bar graph display. <text> is a text string to be displayed in the title bar of the progress indicator. For example,

```
> lister set 121132636 progress 38 'Archiving files...'
```

progress count <count>

This updates the bar graph display in the progress indicator (which must have already been turned on); <count> is the current progress count to be indicated by the bar graph. For example,

```
> lister set 121132636 progress count 4
```

progress name <name>

This updates the filename display in the progress indicator. The filename is displayed above the bar graph. For example,

```
> lister set 121132636 progress name 'myfile.txt'
```

separate <method>

Sets the separation method for this lister. The list is rearranged immediately, but the display will not be updated until you execute a lister refresh command. See the *lister query* section for the separation keywords recognised. For example,

```
> lister set 121132636 separate mix
```

show <pattern>

Sets the show pattern for this lister. The pattern is applied immediately but the display is not updated until you execute a lister refresh command. For example,

```
> lister set 121132636 show '#?.c'
```

sort <method>

Sets the sort method for this lister. The list is resorted immediately, but the display will not be updated until you execute a *lister refresh* command. See the *lister query* section for the sort method keywords available. For example,

```
> lister set 121132636 sort date
> lister set 121132636 sort filetype
```

source [lock]

Makes this lister the source. If you specify the lock keyword, it will be locked as a source. For example,

```
> lister set 121132636 source lock
```

title <string>

Sets the title for this lister (the title displayed in the lister title bar). The title bar display will not be updated until you execute a *lister refresh full* command. The old title is returned in **RESULT**. For example,

```
> lister set 121132636 title 'hello'
--> RESULT
> lister set 121132636 title
--> hello
```

toolbar

This command changes the toolbar that is used in this lister. For example,

```
> lister set 121132636 toolbar Ram:custom_toolbar
```

visible <state>

Sets the visible status for this lister. By default, listers are visible when they are created. If you set this state to 0 or off, the lister will disappear from the display, until you make it visible again. For example,

```
> lister set 121132636 visible off
> lister set 121132636 visible 1
```

- **lister select** <handle> <name> <state>

This command changes the selection status of an entry in the specified lister. <name> is either the name of the entry, or #xxx (where xxx is a number) to specify the cardinal number of the entry. <state> is the desired selection status (0 or *off* for off, 1 or *on* for on). If <state> is not given then the state of the entry is toggled. The display is not refreshed until you execute a lister refresh command. The previous selection state of the entry is returned in **RESULT**. For example,

```
> lister select 121132636 ENV on
--> off
```

- **lister wait** <handle> [**quick**]

This command causes the rexx script to wait for the specified lister to finish whatever it is doing. Because Opus 5 multitasks, all rexx commands (like *lister read*, or *lister new*) will return immediately, even if the lister has not completed its task. This command will force the script to wait until the lister goes non-busy. If the lister is not in a busy state when this command is called, the program will wait for up to two seconds for it to go busy, otherwise this call is aborted. If the *quick* keyword is specified, the command will return immediately if the lister is not busy, instead of waiting for two seconds. It would be silly to do *lister set busy 1* and then *lister wait*. For example,

```
> lister read 121132636 'c:'
> lister wait 121132636
```

- **lister iconify** (all | <handle>) <state>

This command causes either all listers or the specified lister to become iconified if state is 1, on, or omitted altogether, and to deiconify if state is 0 or off.

```
> lister iconify 121132636
> lister iconify all off
```

command

The third base command is **command**. This allows you to call the internal commands of Directory Opus 5 from an ARexx script. The commands execute exactly as if they had been run from a custom button or menu.

command [*wait*] [*source* <*handle*>] [*dest* <*handle*>]
 [*original*] *command* [*arguments*]

If the *wait* flag is specified, the command will be run synchronously, otherwise it will return immediately. Ordinarily, commands operate on the current source and destination listers - the *source* and *dest* parameters allow you to specify alternative listers to use.

The *original* flag allows you to run an *original* Opus internal function if the command has been replaced in the command list by an external module (external modules which add commands to Opus override the internal list). This means you could have a module that replaced some Opus commands, did something special in some circumstances, and in others just called the original Opus function.

The *command* parameter is the name of the command, and *arguments* are any optional arguments for the command, as normal. Some examples are:

- > **command all**
- > **command wait copy**
- > **command read s:startup-sequence**
- > **command source 121132636 mkdir name=MyDir noicon**
- > **command original wait delete ram:#?**

ARexx Error Codes

Lister handles are the actual address in memory of the lister structure. Opus 5 will reject any non-valid handles with an **RC** of 10. All commands that return data return it in **RESULT** (with the exception of *dopus* *getstring* and *lister* *getstring*) or a specified stem variable; if an error occurs, the error code is returned in **RC**.

An **RC** of 0 generally indicates that everything is ok. Error codes are:-

1 RXERR_FILE_REJECTED

The file you tried to add was rejected by the current lister filters.



Note that this is not an error, just a warning. The file is still added, it will just not be visible until the filters are changed.

**5 RXERR_INVALID_QUERY
 RXERR_INVALID_SET**

The query/set item you specified was invalid.

**6 RXERR_INVALID_NAME
 RXERR_INVALID_KEYWORD**

The filename, or keyword you specified was invalid.

8 RXERR_INVALID_TRAP

The trap you tried to remove didn't exist.

10 RXERR_INVALID_HANDLE

The lister handle you gave was invalid.

12 RXERR_NO_TOOLBAR

The lister has no valid toolbar.

15 RXERR_NO_MEMORY

There wasn't enough memory to do what you wanted.

20 RXERR_NO_LISTER

A lister failed to open (usually because of low-memory).

Custom Handlers

The custom handler system allows you to specify the name of an external public message port. This port will be sent messages whenever certain things happen that you are interested in. Messages that are sent are properly formatted ARexx messages. An example code fragment to receive a message is:

```
call waitpkt(myportname)/* wait for messages to arrive */

packet=getpkt(myportname)/* get waiting message */
arg0=getarg(packet,0)/* get Argument 0 */
arg1=getarg(packet,1)/* get Argument 1 */
arg2=getarg(packet,2)/* get Argument 2, etc... */

call reply(packet,0)/* reply to the message */
```

Custom Handlers for Listers

A custom handler is "attached" to a lister by calling *lister set <handle> handler* for that lister, giving the name of your message port. Whenever something interesting happens to your lister, the handler will be sent an ARexx message. The handler can be implemented either as a rexx program or as a C program (in which case it must interpret the rexx message itself). Unlike Opus 4, messages sent to handlers do not cause Directory Opus 5 to "hang" until they are replied (although you should try to reply to any messages as soon as possible).



Note that custom handlers for listers are specific only to the cache that is visible in the lister at the time the handler name is set. The same handler port may be used set for multiple caches, and indeed for multiple listers. Note also that message port names are case-sensitive.

The rexx message identifies the type of event, the lister the event happened to, and other pertinent data. The events that you will be notified of are :-

doubleclick

This is a double-click event, and indicates that an item in the lister has been double-clicked on by the user. The message arguments are:-

Arg0 - "doubleclick" (a string indicating the event type)
Arg1 - <handle> (lister handle)
Arg2 - <name> (entry name)
Arg3 - undefined
Arg4 - undefined
Arg5 - <userdata> (if userdata was specified with the *lister addstem* command)
Arg6 - <qualifiers> (string indicating qualifiers pressed
- shift, alt, and control keys)

drop

This is a drag'n'drop event, and indicates that one or more entries have been dropped into a lister. The message arguments are:-

Arg0 - "drop" (event type)
Arg1 - <handle> (destination lister handle)
Arg2 - <names> (filenames)
Arg3 - <handle> (source lister handle)
Arg4 - undefined
Arg5 - undefined
Arg6 - <qualifiers> (shift, alt, and control keys)

The filenames are separated by spaces (if there is more than one), and will be within quotes if the *quotes* keyword was specified for the *lister set handler* command. If the files originated from another Opus 5 lister, Arg3 gives the handle of that lister. If this is the case, and the *fullpath* option was not specified in *lister set handler*, only the filenames (and not their paths) are supplied in Arg2 (you can get the source path using lister query). If Arg3 is null then the drop most likely originated from Workbench, and the names in Arg2 include the full paths.

dropfrom

This is exactly the same as the drop event, except that it indicates a drop from a lister rather than a drop to one. The message arguments are:-

Arg0 - "dropfrom" (event type)
Arg1 - <handle> (source lister handle)
Arg2 - <names> (filenames)
Arg3 - <handle> (destination lister handle)
Arg4 - undefined
Arg5 - undefined
Arg6 - <qualifiers> (shift, alt, and control keys)

Note that appicons can also receive dropfrom events but they have slightly different arguments. They can be distinguished by the word "icon" always present in Arg4. See below for more information.

parent

This event will be received when the *Parent Directory* item is chosen from the lister cache pop-up menu, or whenever the user clicks on the border parent gadget or uses the parent hot key, "/". The message arguments are:-

Arg0 - "parent" (event type)
Arg1 - <handle> (source lister handle)
Arg2 - <path> (path of lister)
Arg3 - undefined
Arg4 - undefined
Arg5 - undefined
Arg6 - <qualifiers> (only shift key)

root

A root event will be received when the *Root Directory* item is chosen from the lister cache pop-up menu, or whenever the user uses the root hot key, ":". The message arguments are:-

Arg0 - "root" (event type)
Arg1 - <handle> (source lister handle)

Arg2 - <path> (path of lister)
Arg3 - undefined
Arg4 - undefined
Arg5 - undefined
Arg6 - <qualifiers> (shift keys)

path

When the user enters a new path in the path gadget of a lister you will receive this message. The arguments are:-

Arg0 - "path" (event type)
Arg1 - <handle> (lister handle)
Arg2 - <path> (path of lister)

reread

The reread event will be sent to your handler when the *Re-read Directory* item is chosen from the lister cache pop-up menu. Its arguments are:-

Arg0 - "reread" (event type)
Arg1 - <handle> (lister handle)
Arg2 - <path> (new path for lister)

active

This event indicates that a cache with a custom handler attached has just become visible. The message arguments are:-

Arg0 - "active" (event type)
Arg1 - <handle> (lister handle)
Arg2 - <title> (cache title)
Arg3 - undefined
Arg4 - <path> (path of the lister)

Arg2 will contain the custom title of the cache that became active, if it has been set with lister set title. If no custom title has been defined, the path string of the cache is returned instead (ie in this case Arg2 will be the same as Arg4).

inactive

This event indicates that the cache this custom handler is attached to is no longer active (visible in the lister). The message arguments are the same as for *active* above, except for a different event type in Arg0. This message is caused by the cache in the lister being changed (either by the user or under rexx control), or even by the lister being closed. Note that you may receive an *active* message for another cache with a custom handler, or even for the same cache, immediately after receiving an *inactive* message.

Trapped Functions

Messages for trapped commands are sent to the lister much like the other messages. The arguments are :

- Arg0 - <command>(name of the command, or "abort")
- Arg1 - <handle>(source lister handle, if any)
- Arg2 - <files>(selected files, if any)
- Arg3 - <handle>(destination lister handle, if any)
- Arg4 - <path>(source path; useful if there's no lister associated with it)
- Arg5 - <args>(user-supplied arguments to the function)
- Arg7 - <path>(destination path; allows you to support the Select Destination requester)

AddStem Pop-Ups

If you have added files to a lister with *lister addstem* and specified your own pop-up menus for the files, you will receive messages when these menus are chosen by the user. The arguments sent are:

- Arg0 - "menu"(string identifies this as a menu event)
- Arg1 - <handle>(lister handle)
- Arg2 - <name>(entry name)
- Arg3 - <id>(ID of the menu item + base ID if specified)
- Arg4 - "file"(string identifying this as a "file" menu event)
- Arg5 - <userdata>(userdata specified via the *lister addstem* command)

Custom Handlers for AppIcons

AppIcons added with the *addappicon* command will also cause messages to be sent. All AppIcon messages have the same arguments:-

- Arg0 - <event>(string identifying the event)
- Arg1 - <id>(ID specified in the *addappicon* command)
- Arg2 - <data>(filenames/ menu ID/other information)
- Arg3 - <handle>(source lister handle - if applicable)
- Arg4 - "icon"(string identifying this as an "icon" event)

These are the events that apply to AppIcons:

doubleclick

This event indicates that an icon has been double-clicked, or has had *Open* selected from its menu.

dropfrom

This is a drag'n'drop event, and indicates that one or more entries have been dropped on this appicon from a lister or elsewhere. The names of the entries are available in Arg2.

snapshot

This event occurs when the *Snapshot* menu item is selected. The current position of the icon is available in Arg2 (as an x,y string). You are responsible for storing this position.

unsnapshot

This event occurs when the *Un-Snapshot* menu item is selected.

removed

This event warns that Opus has quit and the handler code should now clean up and exit.

info

This event occurs when the *Information* menu item is selected.

close

This will occur when *Close* is selected from the pop-up menu.

menu

This event indicates that one of the user-supplied menu items has been selected in the pop-up menu. The number of the menu item will be returned in Arg2.

menuhelp

This event indicates that the help key was pressed while the mouse pointer was over one of the user-supplied menu items. The number of the menu item will be returned in Arg2.

Because of the multi-tasking nature of Opus 5, information custom handlers receive can not be 100% relied on. For example, you may receive an *active* message, but the cache that caused it may have immediately gone *inactive* again. You should therefore check your port is clear of all messages before processing any that have come in, and you should also use the *lister query* command to make sure that things are how you expect them. Also note that listers (unless you have turned busy on) can be closed by the user at any time. An *inactive* message is sent when the lister is closed. To check that a lister is still open, use the *lister query path* command (or any other query command). If the lister no longer exists, RC will contain the error code **XERR_INVALID_HANDLE (10)**. Be aware though that while these possibilities exist, generally they will not cause a problem. For the most part it will only be if the user is "playing around" that weird situations will occur.

ARexx Modules

ARexx Modules are ARexx scripts which are installed in the *DOpus5:Modules* directory. They must have the suffix **.dopus5** to work correctly.

Each ARexx Module can add new internal commands to Opus.

Once both Opus 5 and ARexx have been started on the computer, the *DOpus5:Modules* directory will be scanned for ARexx Modules and each will have its *init* function called. Every ARexx Module must have an *init* function or else it will not work. This is the function that adds the extra commands to Opus.

The scripts are called with 4 or more parameters. The first 4 are always provided - the portname of Directory Opus 5 and the function name, followed by the source and destination lister handles. Any user-supplied arguments to the function will follow.

You can add as many commands as you like. To add commands, use the *dopus command* command. The template is :

```
dopus command <name> program <scriptname>  
                [desc <description>] [template <template>]  
                [source] [dest]
```

command	- the name of the command
scriptname	- filename of the script (this has to be the name of the script as it appears in the <i>DOpus5:Modules</i> directory, without the <i>.dopus5</i> suffix)
desc	- optional command description (for the pop-up command list in editors)
template	- optional ReadArgs style command template (you'll have to do parsing yourself!)
source	- indicates that the command needs a source directory
dest	- indicates that the command needs a destination directory

The *program* field is mandatory, and Opus will run the script name you provide here whenever this function is invoked. The *desc* and *template* fields are optional. If you specify the *source* or *dest*

Chapter Sixteen

keywords (or both), you will be passed the appropriate lister handle(s) as an argument to the script. If these flags are not set, the arguments will still be passed, but will be zero. Any user-supplied arguments to the function will be at the end of the command line; you are responsible for parsing these yourself.

Here is a complete example of an ARexx Module:-

```
/* Example Directory Opus 5 ARexx Module */

parse arg portname function source dest arguments
address value portname
options results

/* Initialise */

if function='init' then do
    dopus command "Test1" program "test-command" desc "'Test
command 1'" template "TEST/S"
    dopus command "Test2" program "test-command" desc "'Test
command 2'" source
    exit
end

/* Test function 1 */

if function='Test1' then do
    dopus request "'Test command 1 received!'" "Ok"
    exit
end

/* Test function 2 */

if function='Test2' then do
    str="Test command 2 received - source handle " | | source
    dopus request str "Ok"
    exit
end
```

This page is left intentionally blank.

Appendix A

Opus 5.5 Environment Variables

Although most of the operation of Opus 5 can be controlled through the Environment and Options settings, some specialised functions are controlled through system environment variables in the ENV: directory. These are:-

dopus/dopus

Stores temporary preferences for the main Opus 5 program.

dopus/UseWBInfo

Forces the icon.module to call the OS WBInfo() routine to do the icon information function. (This only works under V39, and may only work if you have Workbench running). It is designed to let you use patches like SwazInfo.

dopus/ShowUseDatatypesFirst

The show.module uses its own IFF code in preference to datatypes IFF, since datatypes can be slower with more problems. If required, you can force Opus to use datatypes first by setting this variable.

dopus/3DLook

A special flag to provide compatibility with programs such as SysiHack.

dopus/Text Viewer

Stores preferences and information for the Opus text viewer.

dopus/Print

Stores preferences and information for the Opus print functions.

Appendix B - ViewFont

Opus 5 has no built-in font viewer. To solve this problem we have provided the ViewFont program as a separate stand alone program. This normally resides in the DOpus5:C directory. It can be used from within Opus 5 to show a sample of your selected fonts.

The template for the program is

**FONT,SIZE/N,B BOLD/S,I ITALIC/S,
U ULINE/S,PUBSCREEN/K:**

From within Opus 5, the easiest way of using the program is to create a Filetype which recognizes fonts and then calls ViewFont. We have provided such a filetype as part of the standard Opus 5 installation. Take a look at the Filetype to see how this was done.

Appendix C - Compatibility

Directory Opus 5 is written with system compatibility in mind. It is compliant with the Amiga Style Guide principles and will operate without problems on all Amiga Operating Systems from Workbench 2.0 through 3.1.

At the time this manual was written we are aware of a few programs which may cause problems when run with Opus 5.5.

CrossDOS

The version of CrossDOS [*!CrossDOSFileSystem 38.9 (12/06/92)*] which comes with AmigaOS 3.0 appears to have a bug which affects Opus. If you insert a CrossDOS floppy disk, open an Opus lister for it, then remove the disk and insert a different CrossDOS disk, CrossDOS may lock up.

Solutions:-

Install the version of CrossDOS which comes with AmigaOS3.1 or buy CrossDOS 6.

Make sure you close listers for CrossDOS devices before changing disks.

Cybergraphix

Users of Cybergraphix should be aware of the following variables:

CPUP2C: set to 0 (off). When on causes selected icons in buttonbanks to render in the default four system colours.

PLANES2FAST: set to 0 (off). When on causes garbage and slow-down when dragging icons.

NOPASSTHROUGH: set to (0) off if you have problems using DeluxePaint.

24-Bit Datatypes

If you are using the unofficial 24-bit datatypes, the animations in the *About* window of Opus will appear as garbage. This appears to be the fault of the datatype and may be fixed in a later version. Install the official datatypes if it causes other problems.