Upper Disk Tools

for the ***AMIGA***™

Upper Design™

User Guide

Written by
Manuel Lemos

Copyright

Disclaimer

This product is offered for sale **AS IS** with no representation of fitness for any particular purpose. No Warranty of any kind is either expressed or implied. The user assumes all risks and responsibilities related to its use. The material within is believed to be accurate, but the author and ***Upper Design™*** reserve the right to make changes to the software or documentation without notice.

Trademarks

Upper **Disk Tools™**, ***Recovery™***, ***DSBackup™***, ***Upper Design™*** are trademarks of Manuel Lemos in Portugal and in other countries.

AMIGA™,AmigaDOS and the Amiga logo are trademarks of Commodore-Amiga, Inc. in United States of America and in other countries.

MS-DOS is a registered trademark of Microsoft Corporation.

Macintosh is a registered trademark of Apple Computers.

Notice

All the programs and documentation included in this package were developed using only ***AMIGA™*** computers and ***AMIGA™*** specific software.

Acknowledgements

Upper Design wishes to thank to all of those that volunteered to help making this project possible. ***Upper Design*** wishes to thank in particular to Paul Hammant for his continued support and mostly for being the mentor of this project.

Special thanks to Michael Sinz, Carolyn Schepner, Randel Jesup, James Cooper, Douglas Walker, Nico François and Stefan Becker.

Thanks to Sérgio Soares for having drawn the icon images for the programs.

Also thanks to all those that helped in the localization of this package's programs:
Deutsch
Peter Simons (`simons@peti.GUN.de`)
Thomas Ganter (`ganterth@mathematik.tu-muenchen.de`)
Daniel Amor (`amor@student.uni-tuebingen.de`)
Français
Cedric Beust
Nederlands
Paul Kolenbrander (`boinger@myamy.hacktic.nl`)
Slovenski
Jure Vrhovnik (`Jure.Vrhovnik@ijs.si`)
Dansk
Mark Cassidy
Norsk
Eyvind Bernhardsen (`eyvind@lise.unit.no`)
Svenska
Erik Bergersjoe (`d9erikb@dtek.chalmers.se`)

And others translators that will help with localization to others languages not yet available by the time this manual is being printed.

Special fonts design and localization coordination done by:
Daniel Amor
Brain Storm Development
Ludwigstr. 124
70197 Stuttgart
Germany

Installation

It is recommend that the first thing that should be done before installing any ***Upper*** **Disk Tools** applications, is to make a backup copy of this package's floppy disk. Despite this is not a mandatory procedure, the backup copy should taken to be used as the working copy.

To make a backup copy of the ***Upper*** **Disk Tools** floppy disk, just use from the Shell the AmigaDOS `DiskCopy` command.

```
DiskCopy FROM DF0: to DF0:
```

Consultation of the Amiga's manuals for usage of the shell command `DiskCopy` would be essential reading to understand better how to take a backup copy of a floppy disk.

Although every ***Upper*** **Disk Tools** applications can be started directly from the working copy floppy disk, an `Installer` script is provided to let the applications be installed in an hard disk.

To install the ***Upper*** **Disk Tools** applications in an hard disk, from the WorkBench, open a drawer named `Install` in the working copy of ***Upper*** **Disk Tools** floppy disk. In that drawer there several icons named after several language names. To start the installation process, just double-click any of those icons according to your preferred language.

You will be prompted to reply to several questions about the way you want the ***Upper*** **Disk Tools** applications to be installed. If the language of the selected icon to start the instalation process is already supported by ***Upper*** **Disk Tools** instalation script, the questions you have to reply during installation process will appear in that language.

If you want to un-install ***Upper*** **Disk Tools** applications later after installed, you only need to delete a directory named `UDT` that is created during the installation process.

Contents

Recovery

DSBackup

Introduction

Recovery is a tool that enables the user to retrieve files that have been deleted from a disk, or for some reason are unreadable through normal AmigaDOS usage.

After analysing a disk, it determines what can be recovered and lets the user access the files on the disk and save them to another *volume*.

The files that ***Recovery*** can retrieve, are accessed through an AmigaDOS virtual *device* called `REC:` (*volume* name `Recovery:`). This can be accessed through the Workbench, from the Shell or from any tool capable of reading disk based files.

Recovery is completely safe to use as no data on the volume in question will be modified during its use. That means that neither ***Recovery*** nor any other application using the drive will be allowed to modify data through the use of the `REC:` device. If the disk was damaged before, it won't be any worse after using ***Recovery***.

Quick start

Start ***Recovery*** either through the AmigaDOS Shell or from Workbench.

Use the Shell, Workbench or your favourite directory utility program to browse the directory structure of `REC:` *device* (`Recovery:` *volume*). Storage devices that are scanable by ***Recovery*** appear as directories inside the `REC:Drives` directory.

If you are using the Shell, it is recommended that you use the AmigaDOS `Run` command to start ***Recovery*** as a background program. This allows you to analyse the `REC:` *device* from the same Shell window right after ***Recovery*** is launched.

Examining one of the *drive* sub-directories within `REC:Drives` directory for the first time, will start ***Recovery*** *scanning process* for the disk in the drive with that name.

While this is happening a small window will open showing the progress of the scanning procedure. It also shows an estimate of the time remaining for its completion.

Once scanned, files within that disk that were accessible normally through AmigaDOS usage, are listed within a directory named `Files_in_disk`. Deleted files may be found in either `Deleted_files` or `Lost_files` directories.

Lost files are all of those deleted files for which it was not possible to find their respective parent directory. If ***Recovery*** finds any files for which it was not possible to find their file *header block*, they will appear in `Headerless_Files` directory.

From this stage, you can do anything with the files that does not involve writing data to the actual scanned *volume*. So, you can read, copy, execute or even archive them and perhaps using a backup program to take a backup of any files.

When you have finnished all data recovery from the scanned disk, the respective drive can be freed for normal system usage. This will also release all the other resources allocated during the disk *scanning process*.

See *Scanning a disk* and *Trouble shooting - Freeing a scanned drive* sections.

Usage

Recovery can be started from either the Amiga Shell or the Workbench. Whichever way it is started, it can be customised by using certain options.

When started from the Amiga's Command Shell, these options are listed after name of the program, as in this example:

```
Run Recovery DEVICE=REK VOLUME=REKOVER
```

When started from the Workbench the options are retrieved by ***Recovery*** from its icon. Use `Information` item from Workbench's `Icons` menu after having selected ***Recovery***'s Icon. The options are maintained in the list of `Tool Types` in the Information window.

Whether ***Recovery*** is used from the Shell or the Workbench, its options have the same name and achieve the same effect. Note that all options names must be of the form `OPTION NAME=OPTION VALUE`.

Although all options names are long and explicit, they also have shorter alias to save on finger wear and tear. The example above showed how ***Recovery*** will set the device name (see later) to `REK` when started. The short name for device is `DEV`, so `DEV=REK` would also have achieved the same thing.

Some of the options can be controlled, to a limited degree, while ***Recovery*** is running. ***Recovery*** opens a *user interface window* and there are various buttons and gadgets to allow the control of the options. If ***Recovery*** was started from the Workbench, these options can be saved to ***Recovery***'s icon.

Recovery will use defaults if there are any options missing when it is started. Most of the time ***Recovery***'s options will not need to be changed from the default, so they don't have to be specified.

The user interface window

Most of ***Recovery***'s options may be set in the user interface window. Other actions may also be started from this window.

Disk scanning related buttons

For more detailed information about the options related with these buttons, see *Scanning a disk* section.

Search for files tick boxes

These tick boxes that indicate whether ***Recovery*** will search for the indicated respective type files.

See *Scanning a disk* - `NOT_DELETED_FILES, DELETED_FILES, LOST_FILES, HEADERLESS_FILES` options section.

Search pattern text input box

This is a box where text should be entered to specify an AmigaDOS file pattern. ***Recovery*** will compare all files it finds with that pattern as it scans a volume. Only files that match that pattern available for retrieval. If the pattern is blank, then all files will be eligible for retrieval. The pattern cannot exceed 256 characters in length.

See `SEARCH_PATTERN` option section.

Search method cycle button

This button indicates which one of two possible methods ***Recovery*** should use to scan a disk. It can be either a fast search method or a deep search method. The deep search method usually takes more memory and time to complete.

See `FAST_SEARCH` option section.

Skip read errors tick box

This tick box indicates whether ***Recovery*** will automatically skip over damaged disk blocks during the *scanning process*. If the scanned *volume* has many damaged blocks, it is probably best to set this option.

See `SKIP_ERRORS` option section.

Before scanning a disk... cycle button

This cycle button determines the availability of the device being scanned to other applications. Basically it will prevent that the contents of the disk may be changed during the *scanning process*. It can either make the disk act as write protected or make it unavailable for system use.

See *Scanning a disk* - `LOCK_DRIVE` option section.

Scanned drives list view

This list displays all the *drives* that ***Recovery*** currently has analised. Selecting one of these drives and clicking on the *Free drive* button, will cause that drive to be freed.

Free drive button

This button is used to free any drive selected in the *Scanned drivcs* list view. If none is selected, this button will appear disabled.

See *Trouble shooting -Freeing a scanned drive* sections.

Buffers number input box

This box indicates the maximum amount of blocks that ***Recovery*** will buffer in memory at the same time to store the most read disk blocks. This value affects the performance of ***Recovery*** while scanning a disk as well as while retrieving files from the scanned disk.

See `BUFFERS` option section.

Other buttons and menus

Save button or menu item

Use this button or menu item to tell ***Recovery*** to save the current option settings. The options are saved in ***Recovery***'s Workbench icon as tooltypes. Because of this, if ***Recovery*** was not started from Workbench, this option will appear disabled. The next time ***Recovery*** is started from Workbench these options will be used automatically.

Hide button or menu item

This button or menu item option closes the ***Recovery***'s user interface window without quitting the program. Hitting the close window button or the Escape key (**Esc**) also causes ***Recovery*** to hide its user interface window.

The user interface window can be recalled by activating the *hotkey* or instructing ***Recovery*** to show its interface using the Amiga's Commodity `Exchange` program.

See *Commodity - hot-key* section.

Help button and menu item

Use this button or menu option to ask for help. You can also obtain specific help about the function of most of the user interfaces's window buttons and menu items, by pressing the **Help** key and leaving the mouse pointer over the button or menu item about which help is being asked.

See *Technical notes - On-line help support*.

Quit button or menu item

Recovery can be instructed to quit by either using the quit button or menu item. If ***Recovery*** was started from the Shell, it can also be told to quit by issuing a `Break` AmigaDOS command to its Shell process.

When ***Recovery*** is instructed to quit, it first tries to free all scanned drives. It then tries to remove the `REC:` *device* from the system and closes user interface window (if opened).

Recovery sometimes may not exit when requested due to some directories still being *locked* by other applications.

See *Trouble shooting - Device in Use* and *Freeing a scanned drive* sections.

About menu item

Use this menu item to tell ***Recovery*** to display a window requester that shows both the current version and the revision of the program, as well as showing copyright and author information.

Provide icons menu item

This menu option, causes ***Recovery*** to provide icon files for all Workbench drawers that appear without one in the `REC:` device.

See *Technical notes - Workbench support* section.

Low memory setup menu item

This menu option, causes ***Recovery*** to turn off some of its own options so that extra memory can be conserved during the *scanning process*.

It switches off the *scanning* of *not deleted* and *headerless* files as well reducing the amount of disk block *buffers* to a minimum. It also tries to free all scanned drives that are not currently in use.

Holding any shift key while selecting this menu option, will switch these options on again.

Setting these options to make ***Recovery*** use as less memory as possible, usually severely affects disk scanning performance. But it improves the chances of ***Recovery*** being able to succesfully scan a disk in computers short of available memory.

See *Scanning a disk...* and *Trouble shooting - Freeing a scanned drive* sections.

Scanning a disk

The scanning process is the process by which ***Recovery*** analyses a *drive* for files. It is triggered by scanning one of the directories named after the device name of the drive that is meant to be scanned, that is in the directory `REC:Drives`.

You can use any kind of directory tool to do this, including the WorkBench or the the Amiga's Shell.

When started, a scanning progress requester window will appear showing a fuel gauge progress bar and a timer that shows an estimate of the amount of time remaining to the completion of the scanning process.

The scanning process can also be suspended or cancelled via the respectively named buttons on that scanning progress window.

This process can take some time and memory depending upon the amount of files and directories that are found in the *volume*. It can be speeded up or use less memory by changing some of the program's options. See *Search method*.

There are a few options that are directly related with the scanning process. They can be set either in the *user interface window* or as ***Recovery*** arguments.

Search for files

Search for NOT_DELETED_FILES

This option indicates whether ***Recovery*** should look for files that have not been deleted. They may be found together in a directory called Files_in_disk.

These files are the ones that are supposedly accessible through the normal AmigaDOS usage. Because of that, searching for those files, may not be necessary. It is therefore recommended that this option should be turned off to conserve memory.

When the *fast search method* is used, this option is ignored and its tick box will appear shaded in the *user interface window*, because this method won't look for not deleted files by definition. See *Search method*.

Option argument name: NOT_DELETED_FILES
Short name: NDEL
Valid option values: yes, no
Default value: yes
Example: NDEL=no

Search for DELETED_FILES

This option indicates whether ***Recovery*** should look for deleted files while scanning a *drive*. They may be found together in a directory called Deleted_files.

Note that ***Recovery*** may consider a file as deleted, when because of some error, this file is not reachable through normal AmigaDOS usage. This situation may be caused by an unreadable preceding file in the same directory or any of its parent directories.

Option argument name: `DELETED_FILES`
Short name: `DEL`
Valid option values: `yes, no`
Default value: `yes`
Example: `DEL=no`

Search for `LOST_FILES`

This option indicates whether ***Recovery*** should look for deleted files that for some reason it couldn't find all its parent directories. These files are found in a directory called `Lost_Files`.

If many of these files are found, Workbench may take a long time to sort them before they appear listed in a Workbench drawer window.
Option argument name: `LOST_FILES`
Short name: `LOST`
Valid option values: `yes, no`
Default value: `yes`
Example: `LOST=no`

Search for `HEADERLESS_FILES`

This option indicates whether ***Recovery*** should look for files for which *header blocks* couldn't be found, when possible. These files be found in a directory called `Headerless_Files`.

Searching of these types of files is possible only if the scanned disk was formated as an *old file system* disk.

Option argument name: `HEADERLESS_FILES`
Short name: `HLES`
Valid option values: `yes, no`
Default value: `yes`
Example: `HLES=no`

Scanning SEARCH_PATTERN

A search pattern causes ***Recovery*** to only look for files whose names match the selected pattern. The pattern can have wildcards in it like `#?.doc` meaning all files that end in `doc`.

The search pattern only applies to files. Directories are never filtered. If no pattern is supplied, all scanned files will be accessible.

This option can speed up the scan slightly and facilitate easier finding of the deleted and lost files.

It can be used to split the search in several scans with the advantage of using less memory for each scan, than when no pattern was supplied.

Read about file patterns in the AmigaDOS user manual for further information.

Option argument name: `SEARCH_PATTERN`
Short name: `PAT`
Valid option values: any valid AmigaDOS pattern with upto 256 characters.
Default value: no pattern
Example: `PAT=~(#?.info)`

Search method

To scan a *volume*, ***Recovery*** may use one of two search methods: fast search method and the deep search method.

Fast search method

Fast search method usually causes a faster scan of the drive but it may under some circumstances not be as complete as a deeper scan.

This method is based on the idea that if there are files on the *volume* that are deleted, then the disk blocks that they were using, are marked as available in the disk's *bitmap*.

With this information, ***Recovery*** only has to check those blocks that are available for use, to look for deleted files. If those blocks were not reused after those files were deleted, it will be possible for ***Recovery*** to retrieve the files intact.

This method turns out to be a much faster method to search for files, but this way ***Recovery*** will not look for not-deleted files. This is why the *Not deleted* button of the user interface window appears shaded when fast search is enabled.

If for some reason ***Recovery*** is not able to find a valid *bitmap* for the *volume*, the user is prompted to indicate whether the deep search method should be used or the whole process should be cancelled.

Deep search method

When using this method, ***Recovery*** will look for files in every block of the *volume* being scanned.

Scanning a *volume* using this method, takes more time to complete and it uses more *memory*. Anyway, this is the method to use when searching for files that although were not deleted, for some reason they are not accessible through normal AmigaDOS usage.

This may happen if any of the file's parent directories are either corrupt or unreadable.

Option argument name: `FAST_SEARCH`
Short name: `FAST`
Valid option values: `yes, no`
Default value: `yes`
Example: `FAST=no`

LOCK_DRIVE

This option selects what ***Recovery*** will do before scanning a disk in a drive. It can make the disk either unavailable for system use or make it act as write protected.

Recovery has to do one of these two actions, to ensure the integrity of the contents of the disk while ***Recovery*** is scanning it. This way, any attempt to write any data on it by the system is prevented.

Make the disk unavailable for system use

Making the disk unavailable for system use, means that no data can be read from or written to it. Essentially the operating system thinks the media has been removed from its drive (even hard disks).

This option is meant to ensure ***Recovery*** has complete control over the volume being scanned. Other programs won't be able to access the drive while it is being scanned.

There are no circumstances where ***Recovery*** will initiate or allow data modification of the scanned volume. The scanned drive is always protected from update.

Make the disk act as write protected

This option is meant to allow the drive to both be scanned by ***Recovery*** and used by any other applications. Files can be read or executed by other programs while that disk is still on the scanned drives list.

When the scanned drive that was write protected by ***Recovery*** is freed, the disk must be physically write unprotected, to let the AmigaDOS unlock to succeed.

Read about write protection in the section in the AmigaDOS manual pertaining to the `Lock` command.

Note: because of a feature of AmigaDOS's software write protection, it is necessary that the disk in the drive to be scanned is initially write enabled.

Write protect could have been achieved by moving the appropriate plastic tag on a floppy disk or using the AmigaDOS lock command on any type of disk. Either way the system needs to believe it to be write enabled before it can be made write protected.

Note: It is possible that other programs write data in a disk scanned by ***Recovery*** overriding these protections. This is done by directly accessing the respective disk `trackdisk` like `exec.library`'s device .

Obviously this is not the recommended behaviour for programs, but some utilities like disk formatters and copiers actually do this. Changing the data in a disk this way while ***Recovery*** scanning it, may confuse ***Recovery***.

Option argument name: `LOCK_DRIVE`
Short name: `LOCK`
Valid option values: `yes, no`
Default value: `yes`
Example: `LOCK=no`

`SKIP_ERRORS` while scanning a disk

During the *scanning process*, *device reading errors* may happen because of a possibly damaged sector. At this point, the user is prompted to retry reading that faulty sector or or skip over it.

If a disk is seriously damaged, many of these reading errors may stop the *scanning process*.

This option allows the user to skip all the *reading errors* automatically, avoiding so the error requester to pop-up.

See *Technical notes - Device level reading* errors section.

Option argument name: `SKIP_ERRORS`
Short name: `SERR`
Valid option values: `yes, no`
Default value: `yes`
Example: `SERR=no`

Device options

Recovery mounts an AmigaDOS *device* to provide access to the files it finds in a disk during the *scanning process*. Some parameters of that device can be set through some of ***Recovery***'s options.

`DEVICE_NAME` option

This option determines what name is to be attributed to the *device* mounted by *Recovery*.

It can be any name, but it is recommend a name with only three letters in upper case with an optional unit number to be consistent with other AmigaDOS device names and to save type work to Shell users.

Option argument name: `DEVICE_NAME`
Short name: `DEV`
Valid option values: any sequence upto 30 characters except for characters **:** and `/`.
Default value: `REC`
Example: `DEV=DEL0`

`VOLUME_NAME` option

This option determines what name is to attributed to the *volume* of the *device* mounted by *Recovery*.

It can be any name, but it is recommend a name with any continuous sequence of letters.

Option argument name: `VOLUME_NAME`
Short name: `VOL`
Valid option values: any sequence upto 30 characters except for characters `:` and `/`.
Default value: `Recovery`
Example: `VOL=Retrieval`

`BUFFERS` option

This option determines the amount of buffers AmigaDOS is to attribute to the `REC:` *device* whilst running. Generally speaking the more the better (up to a limit). Note that the more buffers that a *device* uses the less is available in terms of free memory to the system.

The memory size allocated for each buffer is the same as the size of the disk block. Although each disk block size may vary, it usually is 512 bytes large which the byte size of a sector for many types of disks. Since AmigaDOS 3.1 it is possible to have blocks that occupy more that one sector.

The number of buffers for ***Recovery*** can be specified either in the tooltypes, the user interface window and even by using the AmigaDOS Shell command `AddBuffers`.

The minimum amount of buffers is 2. If a smaller amount is specified, ***Recovery*** will use a number of buffers equivalent to the number of disk blocks that fit into track for each disk being scanned.

The amount of buffers used is only allocated before each *device* is scanned. They will only be freed, when the scanned drive is freed also.

If there isn't enough memory to allocate the specified number of buffers, ***Recovery*** will try to allocate a smaller number till it finds an amount for which there is enough memory.

If you want to tell ***Recovery*** to allocate the largest possible amount of buffers in order to improve scanning performance, just specify a very large number, e.g. 1000000.

Consultation of the Amiga's manuals for usage of the shell command `AddBuffers` would be essential reading before changing this parameter.

Option argument name: `BUFFERS`
Short name: `BUF`
Valid option values: 0 or any value higher than 1.
Default value: `0`
Example: `BUF=100`

Commodity options

Commodities are programs that intercept user input. ***Recovery*** also acts as a commodity to determine when its *hot-key* was pressed. When the user presses the *hot-key*, ***Recovery***'s user interface window will pop-up.

Recovery works in the same way as any system commodity. This means that it has options with standard names for commodity options. It can also be controlled through a commodity window interface which is actually the same as the ***Recovery***'s user interface window.

These options control the way that ***Recovery*** interfaces with the Commodities part of the Amiga's operating system.

`CX_POPUP` - Commodity pop-up option

This option determines whether ***Recovery*** will show its window when started. *yes* will cause the window to appear and *no* will cause ***Recovery*** to run in hidden mode when started.

Option argument name: `CX_POPUP`
Short name: `POP`
Valid option values: `yes, no`
Default value: `yes`
Example: `POP=no`

`CX_POPKEY` - Commodity pop-up key

This option determines which *key* will cause ***Recovery*** to show its commodity window interface. This may cause the window to open if previously closed or to come to the front if previously covered by other windows or not being displayed in the current front screen.

The *key* is usually a combination of keys. This can be changed if it clashes with another commodity *hot-key* or if a different combination of keys is preffered.

If an invalid combination is specified for the pop-up key, ***Recovery*** will revert to its default pop-up key without issuing an error message to the user. The pop-up key that has been used by ***Recovery*** is shown in the title of the user interface window.

If for some reason you forget what was the *hot-key* used to pop-up the commodity window interface, you can use the Amiga system's commodity `Exchange` tool to signal ***Recovery*** to show its interface or simply run ***Recovery*** again.

Consultation of the Amiga's manuals will yield further insight into the usage of *hot-keys*.

Option argument name: `CX_POPKEY`
Short name: `KEY`
Valid option values: any valid combination of keys.
Default value: `control alt r`
Example: `KEY=lcommand shift 0`

`CX_PRIORITY` **- Commodity priority**

Commodity programs are notified about user input by the system. All the commodities running are notified one after another acording to an order. That order is determined by each commodity's priority value.

This options determines ***Recovery***'s own commodity prority value. Commodities with higher priorities are able to intercept input before others with lower priorities.

Commodities like ***Recovery*** filter the kind that they are meant to intercept. In ***Recovery***'s case it is its *hot-key*. Commodities with lower priority will never know about same kind of input.

Consultation of the Amiga's manuals will yield further insight into the usage of *priorities*.

Option argument name: `CX_PRIORITY`
Short name: `CX_PRI`
Valid option values: any value between -127 and 128
Default value: `0`
Example: `CX_PRI=10`

System options

Recovery has several options to control the way it interface to the operating system.

Locale `LANGUAGE`

Recovery is fully localized, and it is ready be used in any of the supported languages. This means that the all the text that the program displays can appear in several different languages.

Use the Locale system preference editor to select the default languages. If ***Recovery*** doesn't yet support the selected language, the default built-in language will be used.

System localization support was only introduced in AmigaDOS V2.1. This `LANGUAGE` option is meant only to specify the language to be used when system localization support is not available. Otherwise this option will be ignored.

See *Technical notes - Locale* `LANGUAGE` *support*.

Option argument name: `LANGUAGE`
Short name: `LANG`
Valid option values: any language of the currently supported
Default value: no language, built-in default is used (english).
Example: `LANG=deutsch`

`PUBSCREEN` - Public screen

This option tells ***Recovery*** the name of a public screen where its windows will be opened.

If no name is supplied for this option or for some reason it was not possible to open a window in that screen or there is no screen with the suplied name,

Recovery will open its windows in the default public screen which is usually the Workbench screen.

Option argument name: `PUBSCREEN`
Short name: `SCR`
Valid option values: any currently opened public screen name
Default value: no public screen name, Workbench screen is used
Example: `SCR=My public screen`

`TOOLPRI` - Tool task priority

The task priority value is used to arbitrate in a multi-tasking system like Amiga's, which task preferencially gets to use the CPU. This means that if a task has a certain priority, all the remaining tasks that have lower priority will have to wait till that first task ends its job or goes to sleep.

Applications usually run with priority 0. If you set this option to an higher value, ***Recovery*** will have most of the time the CPU preference over tasks with lower priority till goes to sleep.

Saying that a task goes to sleep, means that it will be waiting for the system to send it a signal telling it that some event that it was waiting for has just happened. This is the case when for instance, a program is waiting for the user to act upon the user interface window or some data to be read from a disk.

This options determines what priority ***Recovery*** task will run at. It is not very important and was only provided to be consistent with other tools that have the same option. Since the default for the priority is already 0, it is not important to change it.

Option argument name: `TOOLPRI`
Short name: `PRI`
Valid option values: any value between -128 and 127
Default value: `0`
Example: `PRI=20`

Outstanding features

Recovery never modifies the disk being scanned

Recovery is absolutely safe to use as it never alters data on the *volume* being scanned, nor allows any other application to modify data on the *volume* while ***Recovery*** is scanning it.

This was done to both inspire trust in the user and provide failsafe access to a possibly damaged disk.

See *Scanning a disk - Lock drive* section.

Use your favourite file management tool

The main advantage of ***Recovery***, is its ability to act as an intermediate *device*, between any standard Amiga application and the actual storage units.

Since this is achieved using a virtual AmigaDOS *device*, that provides access to all the scanned files in a transparent way, the user can still do any thing with those files except for write actions. Files can be read, copied or even executed in a transparent way, as from a normal *volume*.

This facilitates the use of the Workbench, the Shell or any kind of directory utility, for the exploration and recovery of files. Other programs can co-operate in the file recovery process, e.g. backup programs which allow very large files to be recovered to anywhere else, very easily.

Using ***Recovery*** and a hard disk backup program it would be possible to safely recover a file that is larger than any one floppy disk. This is possible as most backup programs will split the file over many floppies.

Another important consequence of this, is that it is possible (and very easy) to recover files or even whole directory structures to another computer in the computer's network. Of course, both computers have to be in the same network.

Support for the newer filesystems

Recovery also retrieves data from the newer versions of the filesystem like International and Directory Cached. This allows disks formated in any of the newer file systems, to be read in a transparent way, even if the version of the system on which ***Recovery*** is being run, does not support those file systems.

Recovery also supports and scans disks which use multiple sectors per block. This feature was introduced in AmigaDOS version 3.1 (V40).

See *Technical notes -Scannable disk types* section.

Headerless files can be recovered

Recovery can also search for files which their *headers blocks* had been wiped or are unreadable because of some error. Headerless files can be scanned for, if the *file system* used in the disk is one of the versions of the old *file system*. This includes the newer variants International and Directory Cached.

See *Scanning a disk - Search for headerless files* section.

Scanning progress is shown

While a disk in a drive is being scanned, ***Recovery*** always gives the user an idea of how much of the disk has already been scanned by means of a fuel guage. There is also a percentage indicator and a timer that shows an estimate of the remaining time to the end of the scanning procedure.

See *Scanning a disk* section.

Fast search method

Recovery can use when possible, a *fast search* method that only searches for files in a disk's free blocks, where the deleted files are most likely to be found.

See *Scanning a disk - Search method* section.

Very low disk access overhead

Recovery *buffers* the most frequently read blocks, while both scanning a disk or reading file data from it. This reduces to a minimum the disk access and preserves memory.

See *Scanning a disk - Device options Buffers option* section.

Drives being scanned can still be available for use

Recovery can optionally be set to not inhibit the use of a device while scanning is in progress. Though this still precludes writing data to that device, reading from it can continue as if ***Recovery*** had never been started.

See *Scanning a disk - Lock drive* section.

Recovery can run under arduous memory conditions

Options can be set before scanning to cause ***Recovery*** to use the minimum of available system memory.

If the computer has enough memory, the options can be set to make the scanning process faster.

See *Technical notes - Memory usage* section.

On-line Help

Recovery provides on-line help to most of its window buttons and menus. Help buttons, also appear in requesters where an important decision has to be made.

See *Technical notes - On-line help support* section.

Localization

Recovery is fully localized even when run in system with earlier than Amiga OS 2.1. So it is ready be used in any country with any of the supported languages.

See *Technical notes - Multi language support* section.

Limitations

Recovery can only retrieve files from *track based disks* like floppies, hard disks, etc... but not from sequentially read storage devices like tape streamers, nor virtual dynamic *devices* like the Amiga's `Ram Disk`.

Recovery doesn't try to retrieve data from corrupt portions of a disk. If while scanning a disk, ***Recovery*** encounters an unreadable block, a requester will open on screen. Via this requester the user can instruct ***Recovery*** to skip the faulty block or abort the scan.

Recovery cannot retrieve deleted files when AmigaDOS has reused the space they occupied on the disk. If a file is accidentally deleted then ***Recovery*** should be started without delay!

In disks formated with *old file system,* file blocks have redundant information, that makes possible to double check if that block really belongs to the file as well the data in that block is corrupt. In disks formated with *fast file system*, each block is completely filled with the file data and so this redundant information is no longer available. Because of this, it is not possible to check if a *fast file system* disk block was corrupted. This is also the reason why ***Recovery*** can't find any *headerless files* in *fast file system* formated disks.

Recovery uses the available memory to store information about scanned files. There can be problems during the *scanning process* if there are too many files and too little memory to store information in. In this scenario, ***Recovery*** can be set to conserve as much memory as possible.

See *User interface window - Low memory setup* section.

Recovery currently ignores any scanned file or directory link blocks, as these are just blocks that contain reference to the real file or directory *header block* that is somewhere else in the disk.

Recovery also doesn't try to correct a problem with Workbench, still present in all versions upto 3.1, where it doesn't check if a directory really exists when it finds its drawer icon file alone.

Trouble shooting

Q. What should I do when ***Recovery*** says that the disk scanning operation failed with a `Not enough memory available!` message?
A. See the *User interface window - Low memory setup* and *Technical notes - Memory usage* sections.

Q. How do I free a drive when I have finished scanning it with ***Recovery***?
A. See the *Freeing a drive* section.

Q. What should I do if ***Recovery*** says that the drive I want to free is still being used?
A. See the *Device in use* section.

Q. How do I quit ***Recovery***?
A. See *User interface window - Quit button and menu item* section.

Q. What should I do if I forget the hot key used to pop-up the ***Recovery*** commodity interface?
A. See *Usage* - `CX_POPKEY` - *Commodity pop-up key* section.

Q. What should I do if ***Recovery*** says that the disk in the drive is write protected?
A. See *Usage* - `LOCK_DRIVE` section.

Q. What should I do ***Recovery*** says that it couldn't find a valid disk type identifier?
A. See the *Technical notes - Scannable disk types* section.

Q. What should I do if it seems there isn't any deleted files although the `Deleted_files` or `Lost_files` Workbench drawers still appear?
A. Sometimes a file or a directory may have been deleted but its icon file may not exist or simply might not appear as deleted. Make sure that the Workbench shows all files in the current drawer you are examining. See your Workbench manual.

In case of low memory...

If your computer runs out of memory during the scanning process, you can gain extra memory by freeing any previously scanned drives. See about *free drive* interface function.

If you still encounter memory problems, you can switch off some search options to again decrease the memory that ***Recovery*** will need. You can do this easily by selecting (from the user interface window options menu) the *Low memory setup* menu option. If ***Recovery*** still has problem completing a scan due to the lack of available memory then you can always split the retrieval of files in two or more parts, by using different *search file patterns* each time the disk is scanned.

For instance you can set the search pattern to `[A-L]#?` for the first scan (and recover any necessary files) and for a second scan you can use the opposite search pattern to recover any remaining files with `~([A-L]#?)` as search pattern.

Freeing a scanned drive

When data recovery is complete, the drive can be freed for normal system usage by one of three possible methods:

- In the user interface window, select the respective drive in the *Scanned drives* list view, and then push the *Free drive* button.

- In the `REC:Scanned_Drives` directory, delete the file with the same name of drive to be freed.

- *Quit **Recovery***.

Recovery can only free a drive when it is no longer being used by other applications through `REC:` *device*.

Make sure if the scanned disk was software *write protected* by ***Recovery*** before it was scanned, it is then physically unprotected or else the system won't unprotect it.

If ***Recovery*** is being run on an Amiga running Amiga OS 3 or higher, and the system encounters a low memory situation, ***Recovery*** will try to free all the scanned drives that are not being used without informing the user. Using the *low memory setup* menu item, also makes ***Recovery*** to try to free all scanned drives.

See *Scanning a disk - Lock drive*, *Device in use* and *User Interface window - Low memory setup* sections.

Device In Use

Any program that has a *lock* on the directory created by ***Recovery*** in `REC:Drives` will prevent that drive being freed.

The Workbench via its open windows, the Shell via its current directory, or other programs that can list a, read from, or save to a directory will *lock* the last one used. This is sufficient to stop ***Recovery*** freeing a drive.

All windows pertaining to `REC:Drives/xxx` where `xxx` is the drive in question and below must be closed. All Shell processes must be set to have another device or volume as their current directory or must be quit. The same is true of other applications. If in doubt then quitting applications you think are locking the drive should free all the locks on it.

It is possible that a faulty application may retain a *lock* for file or a directory from `REC:` device after it has exited or crashed. This will effectively prevent Recovery from quitting.

Technical notes

Workbench support

To support Workbench, ***Recovery*** provides icons for directories and files that it creates in the `Recovery:` *volume* .

This facility can be controlled with the option `PROVIDE_ICONS`. This option can be also set in the user interface window `Options` menu.

Recovery searches through directories for these icon files in the following order: `PROGDIR:Icons/`, `ENV:sys/`, `ENVARC:sys/` where `PROGDIR:` is the directory path of ***Recovery*** program file.

Recovery tries to find icons with the names `def_Disk.info` for the *volume* icon, `def_Drawer.info` for drawer icon and `def_Project.info` for project icons.

If no icons files where found in the above directories then ***Recovery*** will use built in icon images.

Multi language support

Recovery supports language localization using Amiga standard methods, in all supported versions of Amiga operating system.

Although localization was only introduced in Amiga OS version 2.1, localization is still available if ***Recovery*** is run within earlier versions of the OS. In this case, the user's preferred language must be specified with the `LANGUAGE` option, or else the built-in default language will be used.

For each of the supported languages, there is the respective catalog file in a sub-directory in the same directory where ***Recovery*** is started. The path is: `PROGDIR:Catalogs/`**`LANGUAGE`**`/Recovery.catalog.`

The **`LANGUAGE`** directory name is the same as the ***Recovery***'s `LANGUAGE` option or the selected locale language system preferences. If for instance the `LANGUAGE` was set to `deutsch` then the corresponding catalog file must be found in either the following paths:

```
PROGDIR:Catalogs/deutsch
LOCALE:Catalogs/deutsch
```

The installation procedure doesn't put the language catalog files in this last path.

See *System options - Locale language*.

On-line help support

Recovery uses AmigaGuide hypertext system to provide context dependant on-line help. On-line is also localized.

Besides help buttons and the help menu item, it is possible to have context dependant help by pressing the **Help** key with the mouse pointer over a button, gadget or menu item in the user interface window.

If for some reason AmigaGuide is not available, alternative information windows will appear instead.

AmigaGuide based on-line help support will only work if at least version 33 of `amigaguide.library` is available. Use AmigaDOS command `Version amigaguide.library` to check what if and what version of AmigaGuide is available.

The normal ***Recovery*** installation procedure tries to install the AmigaGuide if it is not currently installed.

Memory usage

Recovery uses an amount of available memory to store information about the scanned files and directories on a drive. The amount of memory used is more or less in proportion to the size of the disk.

It is recommended that the user should partition the hard disk in such a way as to ease the later use of ***Recovery*** if a file has accidentally been deleted.

Although ***Recovery***'s memory usage depends mostly on the number files and directories the user has stored in a disk, partitioning a hard disk into smaller, more manageable, chunks will limit the number of files and directories that can be fitted into each partition and hence speed up the scanning process taking less memory to hold the scanned disk's directory tree.

Most of the options available through the ***Recovery*** user interface window, are meant to allow the user to limit program memory usage. Use *Low memory setup* menu option to switch off some options to minimize memory usage.

See *User interface window - Low memory setup* section.

If ***Recovery*** is being run in a computer with Amiga OS 3 or higher, and the system is low in memory ***Recovery*** will try to free all the scanned drives that are not being used.

Scannable disk types

Currently, ***Recovery*** is only able to scan disks which are of AmigaDOS type. It determines that disk type by reading some special reserved sectors at the beginning of the disk or partition being scanned.

The type identifier consists of 4 bytes forming a longword, that are the first bytes of those reserved sectors.

An AmigaDOS disk type identifier must have the first 3 bytes with the letters 'D', 'O' and 'S' respectively. The last letter is used to determine the type of the *file system* used in the disk.

In general, if the number in this fourth byte is even, the *file system* is the old file system, if it's odd it is the fast file system.

So we have:
'D', 'O', 'S', 0 - Original old file system
'D', 'O', 'S', 1 - Original fast file system
'D', 'O', 'S', 2 - International old file system
'D', 'O', 'S', 3 - International fast file system
'D', 'O', 'S', 4 - Directory cached old file system
'D', 'O', 'S', 5 - Directory cached fast file system

If the first reserved sector is unreadable or the disk type identifier was none of those shown above, a requester will appear, prompting the user to indicate whether old file system or fast file system should be assumed as the disk file system.

Both could be tried, but only one is right. If the wrong one is chosen, files may appear as corrupted. In that case, the disk should be rescanned, but the other file system should be selected when asked again.

If ***Recovery*** says that an unknown disk type identifier was found, it is not very likely that the disk is an AmigaDOS disk and most probably no files will appear. This may still occur though, as the first disk block may have been corrupted but not the rest of the disk.

Device level reading errors

Recovery uses Amiga `exec.library` functions to read each device to be scanned. It is assumed that each disk or partition's associated Exec device behaves like the `trackdisk.device`.

All the device reading error numbers that ***Recovery*** reports, are also assumed to mean the same as if the drive being access is like a `trackdisk.device` based floppy.

The meaning of those errors is listed here as quoted from the programming include files `<devices/trackdisk.h>` and `<exec/errors.h>`.

Name	**no.**	**short description**
`TDERR_NotSpecified`	20	general catchall
`TDERR_NoSecHdr`	21	couldn't even find a sector
`TDERR_BadSecPreamble`	22	sector looked wrong
`TDERR_BadSecID`	23	ditto
`TDERR_BadHdrSum`	24	header had incorrect checksum
`TDERR_BadSecSum`	25	data had incorrect checksum
`TDERR_TooFewSecs`	26	couldn't find enough sectors
`TDERR_BadSecHdr`	27	another "sector looked wrong"
`TDERR_WriteProt`	28	can't write to a protected disk
`TDERR_DiskChanged`	29	no disk in the drive
`TDERR_SeekError`	30	couldn't find track 0
`TDERR_NoMem`	31	ran out of memory
`TDERR_BadUnitNum`	32	asked for a unit > `NUMUNITS`
`TDERR_BadDriveType`	33	not a drive that trackdisk groks
`TDERR_DriveInUse`	34	someone else allocated the drive
`TDERR_PostReset`	35	user hit reset; awaiting doom

Other general Exec device errors

Name	no.	short description
IOERR_OPENFAIL	-1	device/unit failed to open
IOERR_ABORTED	-2	request terminated early [after AbortIO()]
IOERR_NOCMD	-3	command not supported by device
IOERR_BADLENGTH	-4	not a valid length (usually IO_LENGTH)
IOERR_BADADDRESS	-5	invalid address (misaligned or bad range)
IOERR_UNITBUSY	-6	device opens ok, but requested unit is busy
IOERR_SELFTEST	-7	hardware failed self-test

Concepts

Volume

A Volume is a disk inserted into a disk *device*.

For example `Extras4.0` is the name of a volume that can be inserted into a floppy drive.

Another may be `Ram Disk` - the non-permanent simulated disk in the computers memory.

A disk inserted into a disk-drive (floppy or permanently fitted hard drive) will have a volume name as well as a device name.

A volume can be renamed eg `Ram Disk` can be renamed to `Disk In Ram`.

Device

An AmigaDOS Device is a sort of interface capable of having data written to it or read from it.

For example `DF0:` is the name of a device that can receive floppy disks. It can be written to and read from.

Another may be `RAM:` - the non-permanent simulated disk in the computers memory. This too has read and write capability.

A disk inserted into a disk-drive (floppy or permanently fitted hard drive) will have a *volume* name as well as a device name.

Other examples of devices may be:
`HD0:` Where Workbench is held for hard disk users.
`HD1:` Work storage for hard disk users.
`RAD:` A reset proof Ram disk.
`DF1:` A second disk drive (if you have one).
`PRT:` The output only printer device.
`SER:` The serial port (read and write)
`PAR:` Read and write capable parallel device.

Only devices capable of storing data on a *track by track* basis will be able to be scanned by ***Recovery***. Printer ports, parallel and serial devices cannot be scanned. Also, sadly, the `RAM:` device cannot be scanned by ***Recovery*** because it is dynamic in nature. This means that it shrinks and expands depending upon how much data is stored within it.

Remember that although `RAD:` is a virtual device, it is also a *track based device* simulated in the computer memory. So, ***Recovery*** can still retrieve files from `RAD:` as well.

When this document refers to a *drive* it means a device capable of *track based* read and write.

Drive

A Drive is a *device* capable of storing data for later recovery. It can be written to and read from by normal Amiga applications.

Recovery can only scan disks in *track based devices*.

Track based device

A track based device is usually a physical *device* like floppies, hard-drives and CD-ROMs, which are capable of reading or writing data in disks layered in tracks. Usually, the information is divided in sectors inside each track.

Due to versatility of AmigaDOS, it is possible to simulate a track based device to store files for instance in the computer memory like the `RAD:` *device*. Although `RAM:` is also a simulated *device* in the computer memory, ***Recovery*** can't scan it to search for files, because the information stored in `RAM:` is not layered as in disk tracks.

File system

AmigaDOS file system is the system that AmigaDOS uses to manage information of files and directories in a disk.

There are two basic types of the AmigaDOS file system: the *OldFileSystem* and the *FastFileSystem*.

Old File System

OldFileSystem (OFS) is the original Filing system used by the Amiga range of computers.

Fast File System

FastFileSystem (FFS) is the newer type of filing system that was introduced with the release of Amiga OS 1.3. FFS allows more data to be stored on a given disk. The gain in storage capacity is at the sacrifice of an invisible form of file continuity checking. This goes mostly un-noticed to the user, who only sees more storage and marginally quicker file accessing.

The lost continuity checking of FFS means that ***Recovery*** cannot retrieve sections of files without *header blocks*. OFS has no problems with these.

Newer still File Systems

Since AmigaDOS 3, other options have been available for filing systems. Although there are basically based on OFS and FFS, they are more efficient and functional.

Directory cacheing is where storage is sacrificed for directory access speed.

International is where file names cater better for international characters.

Since the release 3.1 of AmigaDOS, it is possible to use more than one sector per disk block. This feature improves the performance of disk operations within directories that have large amounts of files and sub-directories.

Alien File Systems

With latter releases of the operating system, Amiga is able to read and write disks which use different file systems than any of those referred above.

Those file systems were originally used in other types of computers that use MS-DOS or Apple's MacIntosh DOS.

The current version of ***Recovery*** is not yet able to retrieve any data from disks that use those alien file systems.

Lock

Lock is the name that is given to a reference that AmigaDOS uses internally to both locate a file or a directory in a *volume* and to protect it from being overwritten or removed from the *filing system*.

Recovery will only exit (or allow you to *free a drive*) if there are no associated locks on the drive.

An opened file, implicitly sets up a lock, preventing it to be deleted. Therefore, all relevant opened files should be closed to allow ***Recovery*** to either exit or allow a device to be freed.

See *Trouble shooting - Freeing a drive* section.

Disk bitmap

When files are written in a *volume*, its *file system* must have a way to determine which of the disk’s blocks are available for use.

This information is available in a set of special blocks in the *volume* that maps the disk’s free blocks.

Each bit of each byte of those bitmap blocks that is set to ’1’, means that the respective block in the *volume* is free for use. This is why that set of blocks is named bitmap.

When a *volume* is found *unvalidated*, it means that the information present in the bitmap blocks does not correspond to the real usage of the *volume* blocks by the files and directories, meaning the bitmap is not valid.

A *volume* is usually found *unvalidated* when for some reason it was not possible to update the bitmap blocks after a write access to the *volume*. This may happen if the computer is reset or turned off during a an incomplete write or delete operation on the *volume*.

If a *volume* is found *unvalidated*, the *file system* will try to validate by rescanning the whole *volume* directory tree structure to find out which blocks are in use.

The validation process is done automatically if the respective *file system* process finds the *volume unvalidated*. It may take a while depending on the number of files and directories in the *volume.*

The validation process may fail if the disk’s directory tree structure is corrupt or unreadable. At this time the system doesn’t allow any data to be written, updated or deleted from the *unvalidated* disk.

Header blocks

An header block is a block in the volume that the *file system* uses to store attributes of files or directories. For instance, the name, the creation date, protection bits, file comments an the location of each sub block are stored in the header block. Sub blocks can be the list of data blocks for a file or the list of header blocks for a child directory.

If, for some reason, a file header block is unreadable or has been overwritten, the recovery process for files will be harder.

See *Scanning a disk - Headerless files* section.

Introduction

DSBackup is a tool that enables the user to backup information about the structure of an AmigaDOS formated disk. Once saved in another disk, this information may be restored later. That might be the case when because of an accident, the disk structure information present in the disk itself, is corrupted.

The disk structure information may be saved in two forms. It can be a mountlist file or an IFF RDB (`Rigid Disk Block`) file.

A mountlist is a text file that is used by AmigaDOS when mounting a device. An AmigaDOS device can be for instance a disk partition.

A mountlist consists of several keywords with the names of each relevant parameter of the AmigaDOS device, and the respective value to be. Mountlists are interpreted by the AmigaDOS `Mount` command to make the device being mounted available to the system.

An IFF RDB file is a custom IFF file that stores all the information contained in a disk's `Rigid Disk Block`. A `Rigid Disk Block` is a block of information stored in the begining of a disk that can be partitioned.

When the computer boots, the disk's auto-boot driver tries to read the `Rigid Disk Block` information from the disk. If that information is found and is valid (not corrupted), the disk's partitions are mounted and the system can boot from one of the partitions.

If the `Rigid Disk Block` is corrupted because of some accident, booting from the disk won't be possible and the disk data won't be accessible. This is the time to use ***DSBackup*** to restore a previously saved `Rigid Disk Block`.

Usage

Start ***DSBackup*** from either Workbench or Shell. ***DSBackup*** main window opens displaying initially a list of AmigaDOS drives.

Selecting one of the drives and hitting the `Show` button opens another window that displays several parameters of that drive and allows the respective mountlist to be saved to a file.

Switching the radio button in the main window to `Device units` switches the lists to display all the phisycal device units that can hold AmigaDOS formated disks.

Selecting one of the device units and hitting the `Show` button, if that device unit holds a disk that has a valid `Rigid Disk Block`, another window opens displaying several parameter about that `Rigid Disk Block`.From that window you can save the `Rigid Disk Block` to an IFF RDB file or verify if a previously saved `Rigid Disk Block` matches the currently present in that device's unit.

From the main window you can also tell the program to restore a previously saved `Rigid Disk Block` to an IFF RDB file. Each IFF RDB file holds enough information about the device unit from which the `Rigid Disk Block` was saved. Usually only hard-disks have `Rigid Disk Blocks`, but notice that some controllers do not support the Commodore's `Rigid Disk Block` standard format, and so they are not supported by ***DSBackup***.

After loading the IFF RDB in memory, a small window requester opens displaying that device's unit information read from the IFF RDB file. That information can be changed in that requester, if it is the case for instance when the disk was moved to different device unit.

After succesfully restoring the `Rigid Disk Block` into the disk, the user is prompted reboot the computer to let the system recognise the newly restored disk's `Rigid Disk Block.`

Options

DSBackup has several options that can be specified when being started. When started from the Amiga's Command Shell, these options are listed after name of the program, as in this example:

```
DSBackup CREATE_ICONS=no TOOLPRI=5
```

When started from the Workbench the options are retrieved by ***DSBackup*** from its icon. Use `Information` item from Workbench's `Icons` menu after having selected ***DSBackup***'s Icon. The options are maintained in the list of `Tool Types` in the Information window.

Whether ***DSBackup*** is used from the Shell or the Workbench, its options have the same name and achieve the same effect. Note that all options names must be of the form `OPTION NAME=OPTION VALUE`.

Although all options names are long and explicit, they also have shorter alias to save on finger wear and tear. The example above showed how to set ***DSBackup*** to not create icons for files it saves and sets priority of its task to 5. The short name for `CREATE_ICONS` is `ICONS`, so `ICONS=no` would also have achieved the same thing.

DSBackup will use defaults if there are any options missing when it is started. Most of the time ***DSBackup***'s options will not need to be changed from the default, so they don't have to be specified.

Locale `LANGUAGE`

DSBackup is fully localized, and it is ready be used in any of the supported languages. This means that the all the text that the program displays can appear in several different languages.

Use the Locale system preference editor to select the default languages. If ***DSBackup*** doesn't yet support the selected language, the default built-in language will be used.

System localization support was only introduced in AmigaDOS V2.1. This `LANGUAGE` option is meant only to specify the language to be used when system localization support is not available. Otherwise this option will be ignored.

Option argument name: `LANGUAGE`
Short name: `LANG`
Valid option values: any language of the currently supported
Default value: no language, built-in default is used (english).
Example: `LANG=deutsch`

`PUBSCREEN` **- Public screen**

This option tells ***DSBackup*** the name of a public screen where its windows will be opened.

If no name is supplied for this option or for some reason it was not possible to open a window in that screen or there is no screen with the suplied name, ***DSBackup*** will open its windows in the default public screen which is usually the Workbench screen.

Option argument name: `PUBSCREEN`
Short name: `SCR`
Valid option values: any currently opened public screen name
Default value: no public screen name, Workbench screen is used
Example: `SCR=My public screen`

`TOOLPRI` **- Tool task priority**

The task priority value is used to arbitrate in a multi-tasking system like Amiga's, which task preferencially gets to use the CPU. This means that if a task has a certain priority, all the remaining tasks that have lower priority will have to wait till that first task ends its job or goes to sleep.

Applications usually run with priority 0. If you set this option to an higher value, ***DSBackup*** will have most of the time the CPU preference over tasks with lower priority till goes to sleep.

Saying that a task goes to sleep, means that it will be waiting for the system to send it a signal telling it that some event that it was waiting for has just happened. This is the case when for instance, a program is waiting for the user to act upon the user interface window or some data to be read from a disk.

This options determines what priority ***DSBackup*** task will run at. It is not very important and was only provided to be consistent with other tools that have the same option. Since the default for the priority is already 0, it is not important to change it.

Option argument name: `TOOLPRI`
Short name: `PRI`
Valid option values: any value between -128 and 127
Default value: `0`
Example: `PRI=20`

`CREATE_ICONS`

If this option is set ***DSBackup*** will create icons for all the files being saved.

The icon template file for mountlists id def_mountlist.info and for the IFF RDB is def_RDB.info.

DSBackup searches through directories for these icon files in the following order: `PROGDIR:Icons/`, `ENV:sys/`, `ENVARC:sys/` where `PROGDIR:` is the directory path of ***DSBackup*** program file.

Option argument name: `CREATE_ICONS`
Short name: `ICONS`
Valid option values: yes, no
Default value: yes
Example: `ICONS=no`