DirectoryOpus

COLLABORATORS			
	TITLE : DirectoryOpus		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		January 18, 2023	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

DirectoryOpus

1.1 DirectoryOpus 4 ARexx Manual v0.7

Preface

The aim of this documentation is to give you all the knowledge needed to write ARexx-scripts for DirOpus.

Since DirOpus is now distributed under the GPL, and original manuals are very hard to find, I decided to write this manual.

This whole thing was written by somebody who never saw the original documentation for DirOpus. The information herein is provided as is, without ANY warranty.

The commands were 'reverse-engineered' (well, sort of :) and tested. There shouldn't be any wrong information in this manual, but probably I missed some features, keywords or switches I didn't know of.

I would be glad about any additional information (i.e. things that I missed out) or if somebody could provide me with an original manual (though I gave up waiting for this a few months ago).

Christoph Gutjahr
Hermannstr. 41
12049 Berlin, Germany

Korodny@gmx.net

1.2 Introduction

Introduction

This manual is not intended to make you familiar with the basic concepts of DirOpus or to give you hints on clever configurations.

It also assumes some familiarity with ARexx programming. Although it may be possible to figure out how to write ARexx programs for DirOpus just by reading this documentation, and by looking at the sample ARexx scripts, it is much easier if you get a copy of one of the ARexx programming tutorials out there.

Document conventions

The following conventions apply to this document:

{argument} - is a REQUIRED argument

[argument] - is a OPTIONAL argument

<keyword> - is a OPTIONAL keyword

ARexx-Commands are always written in UPPERCASE, just for easier reading.
The case doesn't matter for DirOpus, 'QUIT' is the same as 'Quit'.

Basic informations

The main part of the DirOpus 4 window is occupied by two directory listings. These listings are referred as 'listers' in this manual.

Some commands need to know which lister they should operate on. This information can be given as an (optional) numerical argument: '0' is the left lister, '1' is the right lister.
Most of the time, the default is 'active lister', but don't rely on that!

All commands that accept a filename as an argument, should be provided with a file that is listed in the currently active directory lister. a path shouldn't be given. Otherwise errors may occur.

All functions that normally open up a requester also do so when called from ARexx, as long as the required information is not specified as an argument.

1.3 Starting ARexx scripts

Starting ARexx Scripts

There are two different ways to start ARexx-scripts from within Diropus. In

the edit window where external programms are configured, you can set the cycle gadget to 'ARexx' and type the full path to the script as an argument. The second method is to set the cycle gadget to 'AmigaDOS' and use 'SYS:Rexxc/rx <full path to script>' as an argument.

Method A - the 'inelegant' way

The most common method to start scripts from DirOpus is the following:

```
AMIGADOS: SYS:Rexxc/rx <script.dopus>
```

This method is rather inelegant, as the following steps have to be performed to launch a simple arexx script:

- The batchfile 'T:dopustmp.tmp0' gets deleted if it already exists

- A new batchfile ('T:dopustmp.tmp0') is created and started. The following things are then done by the batchfile:

  - 'DOPUS:C/dopusrt' gets called

  - the 'Failat' command is called

  - the stack is resized by calling the 'stack' command

  - finally 'rx' is called and told to run the script

- after the ARexx script has finished, the batchfile exits

Quite a lot of action, but this is the standard process for calling external CLI commands from DirOpus (the CLI command called in this case is 'rx'). For our purpose, i.e. running an ARexx script, this method has several disadvantages.

It is very slow and fussy. And it equals calling the script from a shell, so Diropus' message port is not the default one (this is a minor problem, see

```
 the ARexx message port
).
```

The one advantage of this method is, that you can easily give arguments to the script when calling it, for example:

```
AMIGADOS: SYS:Rexxc/rx DOPUS:Rexx/testscript.dopus {F}
```

But those arguments have to be parsed by the script, which normally results in bigger code as would be neccessary when retrieving the information directly from DirOpus' database, using the powerfull ARexx commands.

Another problem is, that the maximum length of any argument string given to your script is 256 characters. If the string gets longer (in the above example, this could happen if many files are selected, which results in a very big string that replaces {F}), the argument gets split into several substrings. The script is then called several times, each time with a different substring as an argument. This is a problem if your script has to do some calculations (i.e. count the appearance of something), because those calculations would be restarted every time, ending in wrong results.

```
Method B - the elegant solution

It's just the way the creator(s) of DirOpus intended it to be done. It may
look a bit more complicated at first, but you just have to know the DirOpus
ARexx command set a bit and there's no problem at all:

   AREXX: <script.dopus>

This is much more efficient, as DirOpus just calls 'rx' to run the specified
script. You can't give arguments to a script that is started like that. But
with DirOpus' own ARexx commands, it is very easy to retrieve all the
informations required directly from DirOpus. And DirOpus' message port is
the default port, so there's no need to search for the proper message port.
```

1.4 The ARexx message port

```
                DirOpus' ARexx port

If you wish to send commands to DirOpus from a script that was run from
DirOpus then you don't need to worry about the name of DirOpus's ARexx
message port - your commands will automatically be sent to it.

You can get the name of that port by executing

   portname=ADDRESS()

at the beginning of your script. The variable "portname" will then contain
the name of the default port (which will be DirOpus' port, if your script was
started from DirOpus). This may become important if you need to address
another ARexx message port from within your script and then switch back to
DirOpus:

   /* Just a useless example */
   portname=ADDRESS()
   ADDRESS COMMAND
   delete RAM:T/dopus.tmp0
   ADDRESS VALUE portname
   notify "File deleted!"

Attention: This does NOT work if you are using the
                'inelegant' method
               mentioned in the previous chapter to invoke a script ! In this  ↩
                  case, you
should always give the name of DOpus ARexx port as an argument:

   AMIGADOS: SYS:Rexxc/rx <script.dopus> {Qp}

The script could then parse this argument to get the proper port name:

   /* retrieve portname information */
   parse arg portname
   ADDRESS VALUE portname
```

However, if you want to invoke your scripts from outside of DirOpus, you need to be able to tell ARexx whom you want to talk to.

The Diropus4 ARexx port name is 'DOPUS.x', where 'x' is the invocation count (as DirOpus can be run several times at once). The first is 'DOPUS.1", which is also the most common one. But you can not be sure that this particular message port exists, as the DirOpus with this message port may be closed already.

You may choose to ignore this fact, as this is a very unlikely case. Yet, there's a routine that will find out any port name that DirOpus may have at the moment. Decide for yourself if this problem is important enough for you to use it:

```
ports = SHOW('P')
IF INDEX(ports,'DOPUS.') = 0 THEN DO
   SAY "Dopus is not running at the moment !"
   EXIT 10
END
PARSE VAR ports "DOPUS." num .
portname="DOPUS."||num
ADDRESS VALUE portname
```

This routine will search for the first port starting with 'DOPUS.'. If such a port is found, it will be addressed (the variable 'portname' will contain the name of that port). This way you can be sure that you find a message port if there is one.

1.5 Retrieving informations from DirOpus

Retrieving informations from DirOpus

Results from ARexx commands sent to DirOpus will be returned in the two variables 'RC' and 'RESULT'. This is a standard ARexx mechanism. You have to enable it with the command

```
OPTIONS RESULTS
```

at the beginning of your script.

If a command returns a value or information, the data will generally be returned in the 'Result' variable.

Example:

```
/* which directory is displayed in the left lister ? */

OPTIONS RESULTS

ADDRESS DOPUS.1
STATUS 13 0
dirleft=result
SAY "Directory displayed in left lister: "||dirleft
```

'RC' stands for 'Return Code'. After DirOpus recieved an ARexx command, it will set this variable to one of the following values:

RC= 0 : O.K. No errors occured while processing the command

RC= 1 : Error! Your command could not be processed. This is also returned if the user aborts a requester (i.e. a string requester, see

GetString
) by hitting the 'Cancel' button.

RC= 5 : Unknown command (The command was not recognised by DirOpus).

Sometimes a more meaningfull error code is returned. Diropus knows the following error codes:

RC= 103: Not enough memory

Probable cause:
Not enough memory in your Amiga to carry out the operation.

Recovery suggestion:
Close any unneeded windows and applications, then reissue the command. If it still doesn't work, try rebooting. It may be that you have enough memory but it has become fragmented. It is possible that you may need to add more RAM to your system.

RC= 104: Process table full

Probable cause:
There is a limit to the number of possible processes.

Recovery suggestion:
Stop one or more tasks.

RC= 114: Bad template

Probable cause:
Incorrect command line.

Recovery suggestion:
Consult the documentation for the correct command format.

RC= 115: Bad number

Probable cause:
The program was expecting a numeric argument.

Recovery suggestion:
Consult the documentation for the correct command format.

RC= 116: Required argument missing

Probable cause:
Incorrect command line.

Recovery suggestion:

```
         Consult the documentation for the correct command format.

RC= 117: Argument after "=" missing

         Probable cause:
         Incorrect command line.

         Recovery suggestion:
         Consult the documentation for the correct command format.

RC= 118: Too many arguments

         Probable cause:
   Incorrect command line.

   Recovery suggestion:
   Consult the documentation for the correct command format.
RC= 119: Unmatched quotes

   Probable cause:
   Incorrect command line.

   Recovery suggestion:
   Consult the documentation for the correct command format.

RC= 120: Argument line invalid or too long

   Probable cause:
   Your command line is incorrect or contains too many arguments.

   Recovery suggestion:
   Consult the documentation for the correct command format.

RC= 121: File is not executable

   Probable cause:
   You misspelled the command name, or the file may not be a loadable
   (program or script) file.

   Recovery suggestion:
   Retype the filename and make sure that the file is a program file.
   Remember, to execute a script, either the s bit must be set or the
         EXECUTE command must be used.

RC= 122: Invalid resident library

   Probable cause:
   You are trying to use commands with a previous version of AmigaDOS;
         for example, Version 2.0 commands with Version 1.3 Kickstart.

   Recovery suggestion:
   Reboot with the current version of AmigaDOS.

RC= 202: Object is in use

   Probable cause:
   The specified file or directory is already being used by another
```

```
   application. If an application is reading a file, no other program
         can write to it, and vice versa.

   Recovery suggestion:
   Stop the other application that is using the file or directory, and
   reissue the command.

RC= 203: Object already exists

   Probable cause:
   The name that you specified already belongs to another file or
   directory.

   Recovery suggestion:
   Use another name, or delete the existing file or directory, and
         replace it.

RC= 204: Directory not found

   Probable cause:
   AmigaDOS cannot find the directory you specified. You may have made
   a typing or spelling error.

   Recovery suggestion:
   Check the directory name and reissue the command.

RC= 205: Object not found

   Probable cause:
   AmigaDOS cannot find the file or device you specified. You may have
   made a typing or spelling error.

   Recovery suggestion:
   Check the filename or the device name and reissue the command.

RC= 206: Invalid window description

   Probable cause:
   This error occurs when specifying a window size for the Output
         Window, a Shell, ED or ICONX window. You may have made the window
         too big or too small, or you may have omitted an argument. This
         error also occurs with the NEWSHELL command, if you supply a device
         name that is not a window.

   Recovery suggestion:
   Reissue the window specification.

RC= 209: Packet request type unknown

   Probable cause:
   You have asked a device handler to attempt an operation it cannot
         do. For example, the console handler cannot rename anything.

   Recovery suggestion:
   Check the request code passed to device handlers for the
         appropriate response.
```

RC= 210: Object name invalid

Probable cause:
There is an invalid character in the filename or the filename is too long. Remember, filenames cannot be longer than 30 characters and cannot contain control characters.

Recovery suggestion:
Retype the name, being sure not to use any invalid characters, or exceed the maximum length.

RC= 211: Invalid object lock

Probable cause:
You have used something that is not a valid lock.

Recovery suggestion:
Check that your code only passes valid locks to AmigaDOS calls that expect locks.

RC= 212: Object not of required type

Probable cause:
You may have specified a filename for an operation that requires a directory name, or vice versa.

Recovery suggestion:
Consult the documentation for the correct command format.

RC= 213: Disk not validated

Probable cause:
If you have just inserted a disk, the disk validation process may still be in progress. It is also possible that the disk is corrupt.

Recovery suggestion:
If you've just inserted the disk, wait for the validation process to finish. This may take less than a minute for a floppy disk or up to several minutes for a hard disk. If the disk is corrupt, it cannot be validated. In this case, try to retrieve the disk's files and copy them to another disk. You may have to use DISKDOCTOR.

RC= 214: Disk is write-protected

Probable cause:
The plastic tab is in the write-protect position.

Recovery suggestion:
If you're certain you want to write to that particular disk, remove the disk, move the tab, and re-insert the disk. Otherwise, use a different disk.

RC= 215: Rename across devices attempted

Probable cause:
RENAME only changes a filename on the same volume. You can use RENAME to move a file from one directory to another, but you cannot

```
   move files from one volume to another.

   Recovery suggestion:
   Use the MOVE command instead. Alternatively, use COPY to copy the
   file to the destination volume, and delete it from the source
   volume if desired.

RC= 216: Directory not empty

   Probable cause:
   This error occurs if you attempt to delete a directory that
   contains files or sub-directories.

   Recovery suggestion:
   If you are sure you want to delete the complete directory, use the
   ALL option of DELETE.

RC= 217: Too many levels

   Probable cause:
   You've exceeded the limit of 15 soft links.

   Recovery suggestion:
   Reduce the number of soft links.

RC= 218: Device (or volume) not mounted

   Probable cause:
   If the device is a floppy disk, it has not been inserted in a
   drive. If it is another type of device, it has not been mounted
   with the MOUNT command. It is also possible that you have made a
   typing error when specifying the device name.

   Recovery suggestion:
   Insert the correct floppy disk, check the spelling of the device
   name, mount the device, or revise your MountList file.

RC= 219: Seek error

   Probable cause:
   You have attempted to call SEEK with invalid arguments.

   Recovery suggestion:
   Make sure that you only SEEK within the file. You cannot SEEK
   outside the bounds of the file.

RC= 220: Comment is too long

   Probable cause:
   Your filenote has exceeded the maximum number of characters (79).

   Recovery suggestion:
   Use a shorter comment.

RC= 221: Disk is full

   Probable cause:
```

```
   There is not enough room on the disk to perform the requested
   operation.

   Recovery suggestion:
   Delete some unnecessary files or directories, or use a different
   disk.

RC= 222: Object is protected from deletion

   Probable cause:
   The d (deleteable) protection bit of the file or directory is
   clear.

   Recovery suggestion:
   If you are certain that you want to delete the file or directory,
   use PROTECT to set the d bit.

RC= 223: File is write protected

   Probable cause:
   The w (writeable) protection bit of the file is clear.

   Recovery suggestion:
   If you are certain that you want to overwrite the file, use PROTECT
   to set the w bit.

RC= 224: File is read protected

   Probable cause:
   The r (readable) protection bit of the file is clear.

   Recovery suggestion:
   Use PROTECT to set the r bit of the file.

RC= 225: Not a valid DOS disk

   Probable cause:
   The disk in the drive is not an AmigaDOS disk, it has not been formatted,
   or it is corrupt.

   Recovery suggestion:
   Check to make sure you are using the correct disk. If you know the
   disk worked before, use DISKDOCTOR or another disk recovery program
   to salvage its files. If the disk has not been formatted, use
   FORMAT to do so.

RC= 226: No disk in drive

   Probable cause:
   The disk is not properly inserted in the specified drive.

   Recovery suggestion:
   Insert the appropriate disk in the specified drive.

RC= 232: No more entries in directory

   Probable cause:
```

```
   This indicates that the AmigaDOS call EXNEXT has no more entries in
   the directory you are examining.

   Recovery suggestion:
   Stop calling EXNEXT.

RC= 233: Object is soft link

   Probable cause:
   You tried to perform an operation on a soft link that should only
   be performed on a file or directory.

   Recovery suggestion:
   AmigaDOS uses the Action_Read_Link packet to resolve the soft link
   and retries the operation.
```

1.6 Command reference

```
                Command reference

(commands marked with '***' are not described yet)


                 About

                 AddCustEntry
                 ***

                 AddCustHandler
                 ***

                 AddFile

                 AddIcon

                 Alarm

                 All

                 AnsiRead

                 ARexx

                 Assign

                 Auto

                 Auto2

                 Beep

                 BufferList

                 Busy
```

```
ButtonIconify

Byte

CD

CheckAbort

CheckFit

ClearBuffers

ClearSizes

ClearWin

Clone

Comment

Configure

ContST

Copy

CopyAs

CopyWindow

DateStamp

Defaults

Delete

DirTree

Diskcopy

DiskcopyBG
***

DiskInfo

DisplayDir

DopusToBack

DopusToFront

Encrypt

EndFunction
***
```

```
ErrorHelp

Execute

FileInfo

FinishSection
***

Format

FormatBG
***

GetAll

GetDevices

GetDirs

GetEntry

GetFiletype
***

GetFiles

GetNextSelected

GetSelectedAll

GetSelectedDirs

GetSelectedFiles

GetSizes

GetString

Help

HexRead

Hunt

Iconify

IconInfo
***

Install

InstallBG
***

LastSaved
```

```
LoadConfig

LoadStrings
***

LoopPlay

LPlay

MakeDir

Modify
***

Move

MoveAs

NewCLI

NewShell

NextDrives

NNCopy

NNMove

Notify

None

OtherWindow

Parent

ParentList
***

PatternMatch

Play

PlayST

Print

PrintDir

Protect

Query

Quit

Read
```

```
Redraw

Relabel

Remember
***

RemoveEntry
***

RemoveFile

Rename

Request

Rescan

Reselect

Restore
***

Root

Run

SaveConfig

ScanDir

ScrollH

ScrollToShow

ScrollV

Search

Select

SelectEntry

SelectFile

SetVar

SetWinTitle

Show

SmartRead

Status

StopST
```

```
SwapWindow

Toggle

TopText

UnByte

UnIconify

User1

User2

User3

User4

Verify

Version
```

1.7 About

```
Syntax: ABOUT
Result: -

Brings up the 'About DirOpus' window.
```

1.8 AddFile

```
Syntax: ADDFILE {name size type seconds comment protection_bits blocks
                display}
Result: boolean

Adds a 'dummy' entry to the currently active lister.

name            : name of the entry
size            : size
type            : type (>0=directory, <0=file)
seconds         : Datestamp (measured in seconds since 1.1.1978 0:00)
comment         : file comment
protection_bits : protection bits ('HSPARWED' or a part of it)
blocks          : file size in blocks (not supported)
display         : boolean variable, shall DOpus redisplay the currently
                  active lister (recommended) ?

This function will return a boolean variable, that tells you if the process
was succesful.

Example:
```

```
    addfile testdir 10 1 1000 1 RWED 1 1
```

1.9 AddIcon

```
Syntax: ADDICON [file]
Result: -

Adds an icon to the specified file.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.10 Alarm

```
Syntax: ALARM
Result: -

This function plays an alarm like sound.
```

1.11 All

```
                Syntax: ALL
Result: -

Selects all entries in the currently active lister.

See also:


                 None
                 - deselect all files

                 Select
                 - select files that match a given pattern

                 SelectEntry
                 - select entry by number

                 SelectFile
                 - select/deselect specified files
```

1.12 AnsiRead

```
Syntax: ANSIREAD [file]
Result: -

Loads the specified file into the ANSI-textreader.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.13 ARexx

```
Syntax: AREXX [arexx script/arexx command]
Result: -

Starts an ARexx script or executes an ARexx command. If no arguments are
specified, The user is asked to enter a command/scriptname.
```

1.14 Assign

```
Syntax: ASSIGN [device name]
Result: -

Assigns the logical device name 'device name' to the directory displayed in
the currently active lister. 'device name' should NOT contain a trailing
colon!

If 'device name' is not specified, the user is asked to enter a device name.

Example:

   /* assign example */
   assign test
```

1.15 Auto

```
                Syntax: -
Result: -

This command is just kept for compatability reasons. Use
                 User1
                 -

                 User4
                 instead.
```

1.16 Auto2

```
                Syntax: -
Result: -

This command is just kept for compatability reasons. Use
                 User1
                 -

                 User4
                 instead.
```

1.17 Beep

```
Syntax: BEEP
Result: -

This function plays a beep like sound.
```

1.18 BufferList

```
Syntax: BUFFERLIST
Result: -

This function displays a list of all the directories contained in Directory
Opus' internal buffers. The user may then double-click on one of the
displayed buffers to jump to that buffer immediately, rather than clicking
the arrows to cycle through the buffers one by one.
```

1.19 Busy

```
Syntax: BUSY ON/OFF
Result: -

Will turn the mousepointer's "busy state" on/off.
```

1.20 ButtonIconify

```
                Syntax: BUTTONICONIFY
Result: -

This command will "buttoniconify" DirOpus (see manual/online help for more
information).

CAUTION:

Your script will be halted after this command. Execution will be continued
when DirOpus gets uniconified again !
```

See also:

Iconify

UnIconify

1.21 Byte

Syntax: -
Result: -

This command is just kept for compatability reasons. Use
GetSizes
instead.

1.22 CD

Syntax: CD [path]
Result: -

Specify a new 'Current Working Directory'. This is the directory that will be read, if the user presses 'Return' on a empty directory button.

If [path] is not specified, the command will bring up a directory requester and prompts the user to specify the new current working directory.

See
Status
on how to check what the current working
directory is at the moment.

1.23 CheckAbort

Syntax: CHECKABORT
Result: boolean

This command checks, if the user is pressing both mouse buttons at the same time (to abort a function). Will return '1' if both mouse buttons are pressed at the moment, '1' if not.

Additionally, this command resets the abort flag to zero. If you want to use this command in a script, specify 'checkabort' once at the beginning of your script (to reset the abort flag).

Example:

```
/* checkabort example */

OPTIONS RESULTS
checkabort

DO i=0 to 20
   OTHERWINDOW
   checkabort
   IF result=1 THEN EXIT
END
NOTIFY "You didn't press both mouse buttons!"
```

1.24 CheckFit

```
                Syntax: CHECKFIT
Result: -
```

Will calculate whether all selected files and directories would fit in the destination directory, would they be copied there.

The result of this calculation will be displayed in DOpus' title bar.

If the total size of a directory is calculated already, it won't be recalculated. This may lead to problems when files were added/removed from other programs than DOpus. See

```
                ClearSizes
                on how to remove
```

calculated directory sizes from a lister.

Note: If the destination device has changed, this function will rescan the directories, even if they have sizes displayed for them.

See also:

```
                GetSizes
                - Calculate total size of all entries
```

1.25 ClearBuffers

```
Syntax: CLEARBUFFERS
Result: -
```

This function will clear the contents of all buffers other than the two that are currently displayed. All used memory will be deallocated; this is a good way to free up memory quickly if you have lots of used buffers and are running low.

1.26 ClearSizes

```
                Syntax: CLEARSIZES [lister]
Result: -

This will remove the size displayed for all selected directories in the
specified lister. You may wish to do this if you know the size of the
directory has changed.

If the size of a directory is already calculated, it won't be recalculated
if the functions
                 CheckFit
                 or
                 GetSizes
                 are called.

This may cause problems if files were added/removed from other programs than
DOpus. The 'Clearsizes' function removes displayed directory sizes from the
specified lister, so that they have to be recalculated.
```

1.27 ClearWin

```
Syntax: CLEARWIN [lister]
Result: -

Removes all entries from the specified lister. Results in a blank lister.
```

1.28 Clone

```
Syntax: CLONE [file] [newname]
Result: -

This 'clones' the specified file. In other words: An identical copy of the
file (with a new filename) is created in the same directory. If 'newname' is
not specified, the user will be asked for a new filename for the clone.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.29 Comment

```
Syntax: COMMENT [file] [comment] <RECURSE>
Result: -

This will change the file comment of the specified file to 'comment'. If
'comment' is not specified, the user will be asked to enter a comment.

If 'RECURSE' is specified, selected directories will be scanned recursively.
```

If [file] is not specified, this action will be done for all selected files
in the active lister.

1.30 Configure

Syntax: CONFIGURE
Result: -

This will either load the configuration program ConfigOpus into memory and
run it, or if it is already in memory, invoke it. Consult the manual for
information on the many configuration items.

Attention:

Your script will be halted when the configuration program is active. It will
be continued as soon as the main program is activated again.

1.31 ContST

Syntax: CONTST
Result: -

Continues to play a SoundTracker module that was stopped with
StopST
.
This command may require a certain library (stopus.library) - not
confirmed/denied yet.

See also:

PlayST
- Play SoundTracker modules

1.32 Copy

Syntax: COPY [file]
Result: -

Copies the specified file to the destination drawer.

If [file] is not specified, this action will be done for all selected files
in the active lister.

1.33 CopyAs

```
Syntax: COPYAS [file] [newname]
Result: -

Copies the specified file to the destination drawer under the name 'newname'.
If 'newname' is not specified, the user will be asked for a new filename.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.34 CopyWindow

```
Syntax: COPYWINDOW [lister1 lister2]
Result: -

Displays the contents of 'lister1' in 'lister2'. If no listers are
specified, the content of the active lister will be copied to the inactive
one.
```

1.35 DateStamp

```
Syntax: DATESTAMP [file] [datestamp] <RECURSE>
Result: -

Sets the datestamp of the specified file to the specified time and date. If
date is not specified, the current system time and date will be used.
If 'RECURSE' is specified, selected directories are scanned recursively.

'datestamp' has to be provided in the following format:

'DD-MMM-YY HH:MM:SS'

 DD : Day, numerical
 MMM: Month ('Jan', 'Feb', 'Mar' etc.)
 YY : Year, numerical
 HH : Hours
 MM : Minutes
 SS : Seconds

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.36 Defaults

```
                Syntax: DEFAULTS
Result: -

Sets all configuration informations back to the 'hardcoded' defaults. A
```

requester will pop up if the current configuration settings were changed.

See also:

LastSaved
- Load the last saved configuration file

LoadConfig
- Load a specified configuration file

SaveConfig
- Save the current configuration file

1.37 Delete

Syntax: DELETE [file]
Result: -

Deletes the specified file.

If [file] is not specified, this action will be done for all selected files in the active lister.

1.38 DirTree

Syntax: DIRTREE [lister]
Result: -

This function generates a list of all sub-directories in the directory displayed in the currently active lister, in tree format.

1.39 DiskCopy

Syntax: DISKCOPY [source] [destination(s)] <VERIFY>
Result: -

Copies a floppy disk from 'source' to 'destination'. If 'source' or 'destination' are not specified, the requester of the diskcopy function will be opened.

If 'VERIFY' is specified, DOpus will verify the copy process.

You can specify several destination drives at once, i.e.:

```
DISKCOPY DF0: DF1: DF2:
```

will copy the disk in drive 'DF0:' to both 'DF1' and 'DF2' in one go.

1.40 DiskInfo

```
Syntax: DISKINFO
Result: -

This function brings up a requester with some information about the disk that
is displayed in the currently active lister, including space used and free,
datestamp and number of errors on the disk.
```

1.41 DisplayDir

```
                Syntax: -
Result: -

This command is just kept for compatability reasons (to keep older scripts
working). Use
                 Rescan
                 instead.
```

1.42 DopusToBack

```
Syntax: DOPUSTOBACK
Result: -

If Dopus is running on its own screen, this command will put the Dopus screen
to back.

If Dopus is running on Workbench screen, this command will put the Dopus
window to back.
```

1.43 DopusToFront

```
                Syntax: DOPUSTOFRONT
Result: -

If Dopus is running on its own screen, this command will put the Dopus
screen to front.

If Dopus is running on Workbench screen, this command just activates the
Dopus window, that's all.

If you want to put the Dopus screen/window to front and you are not sure if
DOpus is running on its own screen, it would be best to send both
DOPUSTOFRONT and
                 UnIconify
                 to the program:

   /* Make sure user sees the DOpus window */
```

```
   DOPUSTOFRONT
   UNICONIFY
```

1.44 Encrypt

```
Syntax: [file] [password]
Result: -

Will encrypt the specified file with the specified password. If password is
not specified, the user will be asked for a password.

If the password starts with a - (minus) character, the file will be decrypted
instead.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.45 ErrorHelp

```
Syntax: ERRORHELP [error code]
Result: -

Displays Help information for the given error code. If no error code is
specified, the user is asked to enter one.
```

1.46 Execute

```
Syntax: EXECUTE [file]
Result: -

Will execute the specified file (the function assumes that it is a batch file).

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.47 FileInfo

```
Syntax: FILEINFO [file] [seperator]
Result: STRING

This will return various informations about the specified file. The
informations will be seperated by the specified 'seperator' (defaults to
'SPACE').

The informations returned are:
```

- filename
- filesize (measured in bytes)
- filesize in blocks (not supported yet)
- filetype (>0=directory, <0=file)
- is the file selected (boolean variable)
- datestamp part one (days since 1.1.1978)
- datestamp part two (seconds since 0:00h)
- file comment
- protection bits ('HSPARWED', unset bits are replaced with '-')

1.48 Format

Syntax: FORMAT [device] [name] <VERIFY> <QUICK> <NOICONS>
Result: -

This will format the specified device (without further warnings!) and give it the name 'name'. If 'device' is not specified, the standard format requester will come up.

The options 'VERIFY', 'QUICK' and 'NOICONS' are the same as in the requester.

You can also format several medias at once:

```
FORMAT DF0: Data DF1: DF2: Workdisk
```

This would format the disk in 'DF0:' and name it 'Data:', format the disk in 'DF1:' and give it the default name ('empty') and format the disk in 'DF2' and name it 'Workdisk'.

1.49 GetAll

Syntax: GETALL [seperator]
Result: string

Will create a string which contains the names of all ENTRIES of the currently active lister, seperated by the specified 'seperator' (defaults to 'SPACE'). Result will contain that string.

See also:

GetDirs
- get all directories

GetEntry
- get entry # x

GetFiles
- get all files

1.50 GetDevices

```
Syntax: GETDEVICES
Result: -
```

This function displays a list of all devices, volumes and assigned directories present in the system. The user may then read these devices in by double-clicking on them. He can also select entire devices for use with the hunt and search functions.

1.51 GetDirs

```
                Syntax: GETDIRS [seperator]
Result: string
```

Will create a string which contains the names of all DIRECTORIES of the currently active lister, seperated by the specified 'seperator' (defaults to 'SPACE'). Result will contain that string.

See also:

GetAll
- get all entries

GetEntry
- get entry # x

GetFiles
- get all files

1.52 GetEntry

```
                Syntax: GETENTRY [x]
Result: string
```

Returns the name of entry number 'x'.

See also:

GetAll
- get all entries

GetDirs
- get all directories

GetFiles
- get all files

1.53 GetFiles

```
                Syntax: GETFILES [seperator]
Result: string

Will create a string which contains the names of all FILES of the currently
active lister, seperated by the specified 'seperator' (defaults to 'SPACE').
Result will contain that string.

See also:


                 GetAll
                 - get all entries

                 GetDirs
                 - get all directories

                 GetEntry
                 - get entry # x
```

1.54 GetNextSelected

```
                Syntax: GETNEXTSELECTED [lister]
Result: string

Returns the name of the first selected entry in the currently active lister
(or the specified lister). This entry will NOT be unselected afterwards (if
you don't deselect it yourself, you'll always get the same name)! See

                 SelectFile
                 for informations on how to select/deselect files.

See also:


                 GetSelectedAll
                 - get all selected entries

                 GetSelectedDirs
                 - get selected directories

                 GetSelectedFiles
                 - get selected files
```

1.55 GetSelectedAll

```
                Syntax: GETSELECTEDALL [seperator]
Result: string
```

Will create a string which contains the names of all selected ENTRIES in the currently active lister, seperated by the specified 'seperator' (defaults to 'SPACE'). Result will contain that string.

See also:

GetNextSelected
- get next selected entry

GetSelectedDirs
- get selected directories

GetSelectedFiles
- get selected files

1.56 GetSelectedDirs

Syntax: GETSELECTEDDIRS [seperator]
Result: string

Will create a string which contains the names of all selected DIRECTORIES in the currently active lister, seperated by the specified 'seperator' (defaults to 'SPACE'). Result will contain that string.

See also:

GetNextSelected
- get next selected entry

GetSelectedAll
- get all selected entries

GetSelectedFiles
- get selected files

1.57 GetSelectedFiles

Syntax: GETSELECTEDFILES [seperator]
Result: string

Will create a string which contains the names of all selected FILES in the currently active lister, seperated by the specified 'seperator' (defaults to 'SPACE'). Result will contain that string.

See also:

GetNextSelected

```
                - get next selected entry

                GetSelectedAll
                - get all selected entries

                GetSelectedDirs
                - get selected directories
```

1.58 GetSizes

```
                Syntax: GETSIZES
Result: -

Will scan through all selected directories, calculating the total number of
bytes in each directory, and adds the directory sizes to the list.

The total size of all selected entries will then be displayed in DOpus'
titlebar, including a character ('Y' or 'N') that indicates whether the
selected entries would fit in the destination (would they be copied there)
or not. This is only a very rough indication; to be completely accurate you
should use the
                CheckFit
                command instead of this one.

If the total size of a directory is calculated already, it won't be
recalculated. This may lead to problems when files were added/removed from
other programs than DOpus. See
                Clearsizes
                on how to remove
calculated directory sizes from a lister.
```

1.59 GetString

```
Syntax: GETSTRING '{text} [default]'
Result: string

This command allowes you to prompt the user to input a textstring. This
function will return '0' in RC if the user presses 'Return' or hits the
"OK" Button, '1' if the user hits the "Cancel" button.

{text}     : A string of text to be displayed in the requester, should be
             surrounded by quotes if it containes spaces.

[default]  : The default value of the string; that is, the text you wish to
             initally appear in the field.

Example (Note the quotes around the entire argument string!):

   GETSTRING '"Please enter a Filename:" test.jpg'
```

1.60 Help

```
Syntax: HELP
Result: -

Toggles "Help mode" on/off.
```

1.61 HexRead

```
Syntax: HEXREAD [file]
Result: -

Loads the specified file into the HEX-Reader.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.62 Hunt

```
Syntax: HUNT [text]
Result: -

Will search all selected directories in the currently active lister for a
file called "<text>". Wildcards are possible (both standard AmigaDOS ("?",
"#?") and the asterisk ("*")).

If [text] is not specified, a requester will pop up, asking the user for
a filename.

If the specified file is found, DirOpus will ask the user if it shall go
to the directory where the file is located or continue searching. If it is
told to go to that directory, it will select all files which match the
given pattern.

If DirOpus can't find the specified file, this function will return '205'
("File not found error") in 'RC'.
```

1.63 Iconify

```
                Syntax: ICONIFY
Result: -

Iconifies DirOpus. The script will be continued afterwards !

See also:


                 ButtonIconify
```

UnIconify

1.64 Install

Syntax: INSTALL [device] <CHECK>
Result: numerical

This will make the disk in the specified device bootable. If 'device' is not specified, the requester of the install function will come up.

If 'CHECK' is specified, the existing bootblock will be checked and one of the following values will be returned in 'RESULT':

'0' - no standard bootblock
'1' - standard bootblock
'2' - disk is not bootable
'-1' - Error

Several devices can be installed in one go:

INSTALL DF0: DF1:

1.65 LastSaved

Syntax: LASTSAVED
Result: -

Loads the last saved configuration file. A requester will pop up if the current configuration settings were changed.

See also:

Defaults
- Switch back to default configuration

LoadConfig
- Load a specified configuration file

SaveConfig
- Save the current configuration file

1.66 LoadConfig

Syntax: LOADCONFIG [name]
Result: -

Loads the specified configuration file. DOpus will look for the config file

in 'S:' and 'DOpus:s/' and 'Progdir'.

The specified name can contain a path, Dopus will then look at that path
too.

You can omit the suffix '.cfg', DOpus will add it automatically.

If Dopus is not able to find the specified configfile, it will look for
'DirectoryOpus.CFG' (in 'S:' and 'DOpus:s/') and then for
'DirectoryOpus.defCFG' (in 'Progdir').

If DOpus is not able to find any of those configuration files, it will
restart with the internal default configuration (same will happen if
'name' is no valid configuration file).

A requester will pop up if the current configuration settings were changed.

See also:

Defaults
- Switch back to default configuration

LastSaved
- Load last saved configuration

SaveConfig
- Save the current configuration file

1.67 LoopPlay

Syntax: LOOPPLAY [file]
Result: -

Plays the specified soundfile in an endless loop.

If [file] is not specified, this action will be done for all selected files
in the active lister. In this case, the user can skip the actual file and
play the next one by pressing the left mouse button.

1.68 LPlay

Syntax: -
Result: -

This command is just kept for compatability reasons (to keep older scripts
working). Use
LoopPlay
instead.

1.69 MakeDir

```
Syntax: MAKEDIR [dirname]
Result: -

Creates a dir called 'dirname'. If 'dirname' is not specified, the user will
be asked to enter a name for the new directory.
```

1.70 Modify

```
                Syntax: MODIFY [entry] [value]
Result: -

Queries values from Dopus' internal database. 'entry' specifies the database
entry you want to retrieve and must be one of the following:


                 BankNumber
                 ***

                 ButtonRows
                 ***

                 CopyFlags
                 ***

                 DateFormat
                 ***

                 DefaultTool
                 ***

                 DeleteFlags
                 ***

                 DirFlags
                 ***

                 DisplayLength
                 ***

                 ErrorFlags
                 ***

                 FadeDelay
                 ***

                 Filter
                 ***

                 Font
                 ***

                 GeneralFlags
```

```
***

Helpfile
***

HidePattern
***

IconFlags
***

IconifyFlags
***

ListFormat
***

OutputCmd
***

OutputWindow
***

PortName
***

PubScreen
***

ReplaceFlags
***

ScrClockFlags
***

ScrDepth
***

ScrH
***

ScrW
***

ScreenFlags
***

ScreenMode
***

ScreenName
***

SeparateMethod
***

ShowDelay
```

```
                ***

                ShowFreeFlags
                ***

                ShowPatBits
                ***

                ShowPattern
                ***

                SortFlags
                ***

                SortMethod
                ***

                UpdateFlags
                ***

                ViewPlayFlags
                ***

                WindowSize
                ***

                WindowWH
                ***

                WindowXY
                ***

                WindowXYWH
                ***

See also:


                Query
                - query values from Dopus' configuration

                Status
                - retrieve further informations
```

1.71 Move

```
Syntax: MOVE [file]
Result: -

Moves the specified file to the destination directory.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.72 MoveAs

```
Syntax: MOVEAS [file] [newname]
Result: -
```

Moves the specified file to the destination directory and gives it the name 'newname'. If 'newname' is not specified, the user will be asked to enter a new name for the file.

If [file] is not specified, this action will be done for all selected files in the active lister.

1.73 NewCli

```
Syntax: NEWCLI [specifications]
Result: -
```

Opens a new CLI-Window. If you don't give any further argumens to that command, a CLI window with the attributes defined in the configuration will be opened.

Example:

```
NEWCLI CON:0/11/656/400/DOpus-Shell/close/alt0/21/0/0/
```

1.74 NewShell

```
Syntax: NEWSHELL [specifications]
Result: -
```

Opens a new CLI-Window. If you don't give any further argumens to that command, a CLI window with the attributes defined in the configuration will be opened.

Example:

```
NEWCLI CON:0/11/656/400/DOpus-Shell/close/alt0/21/0/0/
```

1.75 NextDrives

```
Syntax: NEXTDRIVES
Result: -
```

Cycles the 'device' buttons (displays the next six buttons).

1.76 NNCopy

```
                Syntax: -
Result: -

This command is just kept for compatability reasons (to keep older scripts
working). Use
                 CopyAs
                 instead.
```

1.77 NNMove

```
                Syntax: -
Result: -

This command is just kept for compatability reasons (to keep older scripts
working). Use
                 MoveAs
                 instead.
```

1.78 None

```
                Syntax: NONE
Result: -

Deselects all entries in the currently active lister.

See also:


                 All
                 - select all files

                 Select
                 - select files that match a given pattern

                 SelectEntry
                 - select entry by number

                 SelectFile
                 - select/deselect specified files
```

1.79 Notify

```
                Syntax: NOTIFY {text}
Result: -
```

```
Brings up a requester with the text specified in {text}. This requester has
one button labeled "Continue" and a standard Title.

You can have multiple lines in the requester text by putting the "return"
character at the end of lines:

   nl='0a'x             /* return character */
   Notify "first line of Text"||nl||"second line of text"

See also:


                Request
                - a 'YES/NO ?' Requester

                Status
                - how to change the text of requester buttons

                Verify
                - a 'YES/NO ?' Requester with a default text
```

1.80 OtherWindow

```
Syntax: OTHERWINDOW
Result: number of active window (0/1)

Will make the "other" lister (the unactive one) active. The number of the
active lister is returned afterwards.
```

1.81 Parent

```
Syntax: PARENT [lister]
Result: -

Will display the parent directory of the directory currently displayed in the
specified lister.
```

1.82 PatternMatch

```
Syntax: {pattern} {string}
Result: boolean

Returns a boolean variable that indicates if the given pattern matches the
given string.
```

1.83 Play

```
Syntax: PLAY [file]
Result: -

Plays the specified soundfile.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.84 PlayST

```
                Syntax: PlayST [file]

Plays the specified soundfile, interpreting it as a SoundTracker/
NoiseTracker module.

If [file] is not specified, this action will be done for all selected files
in the active lister.

See also:


                 StopST
                 - Stop playing a SoundTracker module

                 ContST
                 - Continue playing a SoundTracker module
```

1.85 Print

```
Syntax: PRINT [file]
Result: -

Prints the specified file.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.86 PrintDir

```
Syntax: PRINTDIR [device] <SIZE> <PROTECT> <DATE> <COMMENT>
Result: -

Prints the currently active directory. 'device' is the output device
(defaults to 'PRT:').

<SIZE>    : print file sizes
```

```
<PROTECT> : print protection bits
<DATE>    : print datestamps
<COMMENT> : print file comments
```

1.87 Protect

```
Syntax: PROTECT <SET> [protection bits] <RECURSE>
Result: -

Sets or deletes the protection bits of all selected files. Protection bits
have to be provided as a string ('HSPARWED' or parts of it). If no protection
bits are provided, a requester will pop up.

Specify 'SET' to set protection bits, otherwise they'll be deleted.

If 'RECURSE' is specified, selected directories will be scanned recursively.
```

1.88 Query

```
                Syntax: QUERY {entry}
Result: Various

Queries values from Dopus' configuration (use the
                 Modify
                 command to
change the configuration). 'entry' specifies the database entry you want to
retrieve and must be one of the following:


  BankNumber          - Button bank currently displayed

  ButtonRows          - Number of button rows visible

  CopyFlags           - Operation/Copy flags:

                          Bit 0 set: also copy datestamp
                          Bit 1 set: also copy protection bits
                          Bit 2 set: also copy comment
                          Bit 3 set: Set archive bit after finishing
                          Bit 4 set: Check dest. free space before starting

  DateFormat          - Operation/Date format flags:

                          Bit 0 set: DD-MMM-YY
                          Bit 1 set: YY-MM-DD
                          Bit 2 set: MM-DD-YY
                          Bit 3 set: DD-MM-YY
                          Bit 4 set: Name Substition (Today, Tomorrow)
                          Bit 5 set: 12 hour clock

  DefaultTool         - ***
```

```
DeleteFlags        - Operation/Delete flags:

                       Bit 0 set: Ask before commencing delete
                       Bit 1 set: Ask before deleting files
                       Bit 2 set: Ask before deleting non-empty dirs
                       Bit 3 set: Ignore 'delete' protection bit

DirFlags           - System/Directory flags:

                       Bit 0 set: Always move to an empty buffer
                       Bit 1 set: Use ExAll()
                       Bit 2 set: Auto diskchange
                       Bit 3 set: Auto disk load
                       Bit 4 set: Double-click/Click-m-click dir read
                       Bit 5 set: Re-read changed buffers
                       Bit 6 set: Search buffers on Parent/Root operation
                       Bit 7 set: Expand pathnames

DisplayLength      - ***

ErrorFlags         - Operation/Error check flags:

                       Bit 0 set: Enable DOS requesters
                       Bit 1 set: Enable Opus error requesters

FadeDelay          - Fade delay (see System/View&Play)

Filter             - ***

Font               - ***

GeneralFlags       - Operation/General flags:

                       Bit 0 set:
                       Bit 1 set: Display info
                       Bit 2 set:
                       Bit 3 set: File double-click
                       Bit 4 set:
                       Bit 5 set: Window slider activate
                       Bit 6 set: Click-m-Click drag

Helpfile           - ***

HidePattern        - Hide pattern (see System/Show pattern)

IconFlags          - Operation/Icon flags:

                       Bit 0 set: Create icon with directories
                       Bit 1 set: Perform all actions on icons as well
                       Bit 2 set: Select icons automatically

IconifyFlags       - System/Clocks...when iconified flags:

                       Bit 0 set: Memory monitor
                       Bit 1 set: CPU monitor
                       Bit 2 set: Date
                       Bit 3 set: Time
```

```
                        Bit 4 set: No window when iconified
                        Bit 5 set: An Appicon when iconified
                        Bit 6 set: Show free space as Kilobytes free
                        Bit 7 set: Text format (free space)= 'C: and F:'

ListFormat {x}      - Selected display items (see Operation/list format),
                      'x' specifies the lister:

                        '0' - Filename
                        '1' - Filesize
                        '2' - Protection bits
                        '3' - Creation date
                        '4' - File comment
                        '5' - File type

OutputCmd           - ***

OutputWindow        - ***

PortName            - Name of Dopus' ARexx port

PubScreen           - name of the PubScreen

ReplaceFlags        - ***

ScrClockFlags       - System/Clocks flags:

                        Bit 0 set: Memory monitor
                        Bit 1 set: CPU monitor
                        Bit 2 set: Date
                        Bit 3 set: Time
                        Bit 4 set:
                        Bit 5 set: Show free space as bytes free
                        Bit 6 set:
                        Bit 7 set: Text format (free space)= 'C: and F:'

ScrDepth            - Screen depth of Dopus' screen (1-8 bits)

ScrH                - Height of Dopus' screen (returns the height of Dopus'
                      window if Dopus isn't running on its own screen)

ScrW                - Width of Dopus' screen (returns the width of Dopus'
                      window if Dopus isn't running on its own screen)

ScreenFlags         - Screen flags:

                        Bit 0 set:
                        Bit 1 set:
                        Bit 2 set:
                        Bit 3 set:
                        Bit 4 set:
                        Bit 5 set:
                        Bit 6 set:
                        Bit 7 set:

ScreenMode          - What screen mode is DOpus using ? Returns either the
                      ModeId, or one of the following:
```

```
                        '1' - Using WB screen
                        '2' - Using a clone of the WB screen
                        '3' - Using other custom public screen

ScreenName          - Name of the screen Dopus is using

SeparateMethod {x}  - Separate Method (see Operation/List format), 'x'
                      specifies the lister:

                        '0' - Mix files and directories
                        '1' - Directories first
                        '2' - Files first

ShowDelay           - Show delay (see System/View&Play)

ShowFreeFlags       - Show free space as... (see System/Directories):

                          Bit 0 set: Bytes free
                          Bit 1 set: Kilo/Megabytes free
                          Bit 2 set: Blocks free
                          Bit 3 set: Percentage of space free

ShowPatBits         - Hide files with hidden bit set (see System/Show
                      pattern)

ShowPattern         - Show pattern (see System/Show pattern)

SortFlags           - Reverse sorting (see Operation/List format)

SortMethod {x}      - Sort method for the listers (see Operation/List
                      format):

                        '0' - Filename
                        '1' - Filesize
                        '2' - Protection bits
                        '3' - Creation date
                        '4' - File comment
                        '5' - File type

UpdateFlags         - Operation/Update flags:

                          Bit 0 set: Update free disk space
                          Bit 1 set: Scroll dir window to follow operations
                          Bit 2 set: Redraw more than a quarter of a page
                          Bit 3 set: Directroy refresh using StartNotify()
                          Bit 4 set: Left-justify filename in status bar
                          Bit 5 set: Operation progress indicator
                          Bit 6 set: Current file progress indicator (Copy)

ViewPlayFlags       - System/View&Play flags:

                          Bit 0 set: Black background between pictures
                          Bit 1 set: Turn audio filter off for play
                          Bit 2 set: 8-bits per gun color
                          Bit 3 set: Loop on double-click
                          Bit 4 set: Text viewer borders
```

```
                        Bit 5 set: Start animation paused
                        Bit 6 set: Pick best screen-mode

  WindowSize          - ***

  WindowWH            - Width and height of Dopus' window

  WindowXY            - Position of top left corner of Dopus' window

  WindowXYWH          - Position, width and height of Dopus' window


Example:

   /* How big is Dopus' window ? */
   QUERY WindowWH
   NOTIFY result

See also:


                Modify
                - Change Dopus' configuration

                Status
                - retrieve further informations
```

1.89 Quit

```
Syntax: QUIT <FORCE>
Result: -

Quits DirOpus. If 'FORCE' is specified, the quit requester won't appear.
Note: The 'You have changed the configuration" requester will always appear.
```

1.90 Read

```
Syntax: READ [file]
Result: -

Loads the specified file into the textreader.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.91 Redraw

```
                Syntax: REDRAW
Result: -

Redrawes the whole screen/window of DirectoryOpus. If Dopus runs on its own
screen, the screen will be closed and reopened. This will make all changes
to the configuration visible (See
                 Status 31
                 for more information).
```

1.92 Relabel

```
Syntax: RELABEL [newname]
Result: -

Relabels the volume that is displayed in the currently active lister.
If 'newname' is not specified, the user will be asked to enter a new name for
the volume.
```

1.93 RemoveFile

```
Syntax: REMOVEFILE {file} [display]
Result: boolean

Removes the specified file from the aktive lister. Returns a boolean variable
that indicates if the file existed and if it could be removed.

The boolean variable 'display' indicates if the displayed directory should be
rescaned afterwards.
```

1.94 Rename

```
Syntax: RENAME [file] [newname]
Result: -

Renames the specified file into 'newname'. If 'newname' is not specified, the
user will be asked to enter a new name for the file.

If no arguments are specified, all selected entries in the currently active
lister will be renamed.

Both 'file' and 'newname' can contain wildcards (usually '*') to specify
patterns (only with selected files):

   RENAME '*.ilbm' '*.iff'

Will change the suffix '.ilbm' to '.iff' - for all selected files !

   RENAME '*' '*.txt'
```

```
Will add the suffix '.txt' to all selected files.
```

1.95 Request

```
                Syntax: REQUEST {text}
Result: boolean

Brings up a requester with the text specified in {text}. This requester has
two buttons labeled "Ok" and "Cancel", aswell as a standard Title.

Returns a boolean variable that indicates which button was pressed
('1'='Ok').

You can have multiple lines in the requester text by putting the "return"
character at the end of lines:

   nl='0a'x             /* return character */
   Request "first line of Text"||nl||"second line of text"

See also:


                Notify
                - a notify-requester

                Status
                - how to change the text of requester buttons

                Verify
                - a 'YES/NO ?' Requester with a default text
```

1.96 Rescan

```
Syntax: RESCAN [lister]
Result: -

Rescans the specified lister.
```

1.97 Reselect

```
Syntax: RESELECT
Result: -

This function will reselect all entries that were selected before the last
operation was initiated. The entries are reselected only if they still exist,
and even if the buffer containing them is not displayed currently.
```

1.98 Root

```
Syntax: ROOT [lister]
Result: -

Displays the root directory of the volume currently displayed in the
specified lister.
```

1.99 Run

```
Syntax: RUN [file]
Result: -

Runs the specified file.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.100 SaveConfig

```
                Syntax: SAVECONFIG
Result: -

Saves the current configuration to the file it was loaded from.

See also:


                 Defaults
                 - Switch back to default configuration

                 LastSaved
                 - Load the last saved configuration file

                 LoadConfig
                 - Load a specified configuration file
```

1.101 ScanDir

```
Syntax: SCANDIR {path} [lister]
Result: -

Reads the directory specified in 'path' into the specified lister.
```

1.102 ScrollH

```
Syntax: SROLLH {lister} {chars}
Result: numerical

Scrolls the specified lister horizontally. 'Chars' indicates how far it
should be scrolled to the left (chars<0) or right (chars>0).

The new horizontal position will be returned.
```

1.103 ScrollToShow

```
Syntax: SCROLLTOSHOW {filename} [lister]
Result: -

Scrolls the active lister (or the specified lister) to show the specified
file (if it is not visible at the moment).
```

1.104 ScrollV

```
Syntax: SROLLV {lister} {chars}
Result: numerical

Scrolls the specified lister vertically. 'Chars' indicates how far it
should be scrolled to the top (chars<0) or to the bottom (chars>0).

The new vertical position will be returned.
```

1.105 Search

```
Syntax: SEARCH [search string]
Result: -

Will search all selected entries for the given search string. If 'search
string' is not specified, the user will be asked to enter a search string.

Directories will be scanned recursively.
```

1.106 Select

```
                Syntax: SELECT [pattern] <NAME>/<DATE>/<PROTECTION> <ONLYFILES>/< ↩
                    ONLYDIRS>
Result: -

Selects files and directories in the currently active lister based upon name
(using wildcards), creation date or protection bits.
If 'ONLYFILES' or 'ONLYDIRS' is specified, only files/dirs are selected.
```

If no arguments are specified, a requester will pop up.

Attention:

'Select' is also a standard ARexx command. ALWAYS enclose this command in quotes, otherwise your scripts won't work (see examples) !

Selection upon name

For selection upon name, all standard wildcards are supported (*,~,? etc..) as well as the AmigaDOS wildcards #? and ?.

Select all entries (files only) with the suffix ".info":

```
'SELECT #?.info NAME ONLYFILES'
```

Select all entries but "background.jpg" and "001.gif":

```
'SELECT ~(background.jpg|001.gif) NAME'
```

Selection upon protection bits

To specify protection bits you use the character codes 'hsparwed'. If you precede a bit with a '-' character, the bit must not be set for the file to be selected.

Select all entries with the 'Read-', 'Write-' and 'Execute-' bit set:

```
'SELECT RWE PROTECTION'
```

Select all entries with the 'Read-' protection bit unset:

```
'SELECT -R PROTECTION'
```

Selection upon date

Selection upon date allows you to specify date ranges, in the form DD-MMM-YY > DD-MMM-YY. You can leave out either of the two dates to specify from the beginning of time, or until the current date. You may also enter times in the form HH:MM:SS as well as dates.

See also:

All
- select all files

None
- deselect all files

SelectEntry
- select entry by number

SelectFile
- select/deselect specified files

1.107 SelectEntry

```
                Syntax: SELECTENTRY {x} {status} [display]
Result: -

Selects/deselects entry number 'x'. 'Status' indicates if the entry should
be selected ('1') or deselected ('0'). Display indicates if display should be
refreshed ('1', recommended) or not ('0').

See also:


                 All
                 - select all files

                 None
                 - deselect all files

                 Select
                 - select files that match a given pattern

                 SelectFile
                 - select/deselect specified files
```

1.108 SelectFile

```
                Syntax: SELECTFILE {file} {status} {display}
Result: -/-1

Selects/deselects the specified entry. 'Status' indicates if the entry should
be selected ('1') or deselected ('0'). Display indicates if display should be
refreshed ('1', recommended) or not ('0').

If the specified file doesn't exist in the currently active lister, result
will set to '-1'.

See also:


                 All
                 - select all files

                 None
                 - deselect all files

                 Select
                 - select files that match a given pattern

                 SelectEntry
```

```
                - select entry by number
```

1.109 SetVar

```
Syntax: SETVAR {variable} {value}
Result: ???

Sets the specified local environment variable to the given value.

???
```

1.110 SetWinTitle

```
Syntax: SETWINTITLE {name} [lister]
Result: -

Replaces the title of the specified directory lister (that usually contains
the name of the displayed volume) with {name}.
```

1.111 Show

```
Syntax: SHOW [file]
Result: -

Tries to show the specified file. The resulting action depends on the user's
configuration.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.112 SmartRead

```
Syntax: SMARTREAD [file]
Result: -

Loads the specified file into the textreader. If 'file' contains characters
that are not displayable as text, the file will be loaded into the
Hex-Reader.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.113 Status

```
                Syntax: STATUS {entry} [x] [SET {y}]
Result: Various

The 'STATUS' command is the workhorse of DirOpus ARexx programs. This
command lets you query the values of DirOpus' internal database, which
gives you access to quite a few internal variables and settings.

{entry}   : Database entry you want to retrieve informations from (see
            below).

[x]       : Most database entries contain informations for more than one
            item. This argument specifies which 'subentry' you are talking
            about. This can be the number of a lister or a directory
            buffer for example.

[SET {y}] : Allows you to change a value by SETTING it to a new value.
            Most STATUS values can be changed (see examples).


Here we go:


STATUS  1     : Path of the current working directory (the directory that is
                read if you press <Return> on an empty directory button).

STATUS  2     : Version number of DirOpus

STATUS  3     : Number of the currently active lister. You can also set the
                active lister ('STATUS' 3 SET 1').

STATUS  4 [x] : Number of FILES in lister [x]

STATUS  5 [x] : Number of DIRECTORIES in lister [x]

STATUS  6 [x] : Number of ENTRIES (files AND dirs) in lister [x]

STATUS  7 [x] : Number of selected FILES in lister [x]

STATUS  8 [x] : Number of selected DIRECTORIES in lister [x]

STATUS  9 [x] : Number of selected ENTRIES (files AND dirs) in lister [x]

STATUS 10 [x] : Total size of all entries in lister [x], measured in Bytes.
                NOTE: If the lister contains directories with unknown size
                (Diropus doesn't display the size next to the name of the
                directory), this function will count them with a filesize
                of "0"! See
                 GetSizes
                 or
                 CheckFit
                 for more
                information.

STATUS 11 [x] : Total size of all SELECTED entries in lister [x], measured
```

```
                in Bytes. See 'STATUS 10' for further notes.

STATUS 12 [z] : The last pattern that was used for the 'SELECT' function
                (either via the
                 ARexx command
                 or the tiny button in the
                center of DirOpus' window, labeled 'S').
                [z] specifies the pattern you want to get, as there are three
                different selection methods (name-based (z=0), creation date
                (z=1) and protection bits (z=2)).
                The specified pattern may also be redefined ('STATUS 12 0
                SET "*.info"')

STATUS 13 [x] : Complete path of the directory displayed in lister [x]. If
                you specify a new directory (i.e. 'STATUS 13 1 SET RAM:'),
                this directory will be read into the specified lister.

STATUS 14 [x] : Name of the volume displayed in lister [x] (without the
                trailing colon!): If the lister displays the directory
                "RAM:Env/Sys", this command will return "RAM". This value
                can NOT be set, use
                 RELABEL
                 instead.

STATUS 15 [x] : Free space on the volume displayed in lister [x], measured
                in bytes.

STATUS 16 [x] : Total size of the volume displayed in lister [x], measured
                in bytes.

STATUS 17 [x] : Complete path of the directory that is buffered in buffer
                [x] (0-99).

STATUS 18 [x] : Name of the volume that is buffered in buffer [x] (0-99).
                See 'STATUS 14' for further notes.

STATUS 19 [x] : Free space on the volume that is buffered in buffer [x]
                {0-99), measured in bytes.

STATUS 20 [x] : Total size of the volume that is buffered in buffer [x]
                (0-99), measured in bytes.

STATUS 21 [x] : Number of the buffer (0-99), that contains the directory
                displayed in lister [x]. Setting this value ('STATUS 21 0
                SET 40') lets you move through the directory buffers.

STATUS 22 [x] : Number of the top line visible in lister [x]. In other
                words: Where does the currently visible section of the
                directory displayed in lister [x] start ?

STATUS 23     : -no longer supported-

STATUS 24     : How many entries can be displayed per lister ? Depends on
                DirOpus' screen- and windowsize.

STATUS 25     : Returns a boolean value that tells you if the configuration
                was changed since it was saved the last time.
```

```
STATUS 26     : Text for the "Okay" button in requesters. Can be changed by
                setting it ('STATUS 26 SET 'AllRight'), but this doesn't
                affect 'Notify' requesters (requesters with only one
                button).

STATUS 27     : Text for the "Cancel" button in requester. Can be changed
                by setting it ('STATUS 27 SET 'No!!!').

STATUS 28     : Returns a boolean value that tells you if DirOpus is
                iconified at the moment.

STATUS 29     : -no longer supported-

STATUS 30     : How's text justified in the title bar ? Returns "0" if text
                is centered, "1" if it is right justified and "2" if it is
                left justified. This value can be set ('STATUS 30 SET 1')

STATUS 31     : Memory Adress of DOpus configuration structure. The
                structure is listed in the file 'Structures.h'. This enables
                you to read and change configuration data.
                Remark: If you change certain configuration settings (like
                screen size for example) the changes won't be effective
                immidiately, this requires a restart or calling the

                 redraw
                 function.
                The following configuration settings can't be changed from
                within ARexx: Number of buffered directories, name of the
                helpfile and the flag that tells if the configuration program
                should be resident.

STATUS 32     : -no longer supported-

STATUS 33     : First filetype. The filetype definitions are saved in memory
                as a chained structure list. This command returns the
                (decimal) address of the first filetype.

STATUS 34     : Number of the button bank currently displayed.


See also:


                 Modify
                 - Change Dopus' configuration

                 Query
                 - query values from Dopus' configuration
```

1.114 StopST

```
                Syntax: STOPST
Result: -
```

Stops playing of SoundTracker modules.

See also:

PlayST
- Play a SoundTracker module

ContST
- Continue playing a SoundTracker module

1.115 SwapWindow

Syntax: SWAPWINDOW
Result: -

Swaps the content of the two directory listers.

1.116 Toggle

Syntax: TOGGLE
Result: -

This function reverses the state of all files and directories in the active directory (ie, all selected entries are deselected, and vice versa).

1.117 TopText

Syntax: TOPTEXT {text}
Result: -

Will show the the specified text in DirOpus' window titlebar.

1.118 UnByte

Syntax: -
Result: -

Just kept for compatability reasons. Use
Clearsizes
instead.

1.119 UnIconify

```
                Syntax: UNICONIFY
Result: -

Uniconifies DirOpus.

See also:


                 ButtonIconify

                 Iconify
```

1.120 User1

```
Syntax: USER1 [file]
Result: -

'User1' - 'User4' are internal commands that can be defined by the user.
This command invokes 'User1' for the specified file. The resulting
operation depends on the user's configuration.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.121 User2

```
Syntax: USER2 [file]
Result: -

'User1' - 'User4' are internal commands that can be defined by the user.
This command invokes 'User2' for the specified file. The resulting
operation depends on the user's configuration.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.122 User3

```
Syntax: USER3 [file]
Result: -

'User1' - 'User4' are internal commands that can be defined by the user.
This command invokes 'User3' for the specified file. The resulting
operation depends on the user's configuration.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.123 User4

```
Syntax: USER4 [file]
Result: -

'User1' - 'User4' are internal commands that can be defined by the user.
This command invokes 'User4' for the specified file. The resulting
operation depends on the user's configuration.

If [file] is not specified, this action will be done for all selected files
in the active lister.
```

1.124 Verify

```
                Syntax: VERIFY [text]
Result: numerical

Brings up a requester with the text specified in [text]. This requester has
two buttons labeled "Ok" and "Cancel", aswell as a standard Title.

If [text] is not specified, the default text ('Do you really wish to
proceed with this operation ?') will be displayed.

Returns a boolean variable that indicates which button was pressed
('1'='Ok').

You can have multiple lines in the requester text by putting the "return"
character at the end of lines:

   nl='0a'x             /* return character */
   Verify "first line of Text"||nl||"second line of text"

See also:


                Notify
                - a notify-requester

                Request
                - a 'YES/NO ?' Requester

                Status
                - how to change the text of requester buttons
```

1.125 Version

```
Syntax: VERSION
Result: -

Brings up the 'Version' window.
```

1.126 Still under construction

```
Under construction !

Sorry, I'm still working on this guide. This node is not implemented yet.
Come back at a later time !  :)
```

1.127 Frequently asked questions

```
Frequently Asked Questions - Still under construction !

Sorry, I'm still working on this guide. This node is not implemented yet.
Come back at a later time !  :)

But if you have got some questions not answered by the rest of this manual,
let me know !
```