

FSYS - *Software*



DFSee HOWTO

Jan van Wijk



Version: 1.2

Sunday, May 23, 2010

Table of Contents

Using the DFSee bootable CDROM.....	3
The DFSee menu system and how to use it.....	5
Navigating the DFSee menu.....	6
Context sensitive help and statusbar description.....	6
Changing the default menu behaviour.....	7
DFSee DFSDISK procedures.....	8
Getting the DFSDISKI.* files.....	8
DFSDISK from the menu inside DFSee.....	8
DFSDISK from the operatingsystem commandline.....	9
Analysis of the results.....	9
CREATE new partitions in freespace areas.....	10
UNDELETE or RECOVER files from HPFS, NTFS or FAT.....	12
IMAGING using RAW or compressed files.....	14
Selecting the object.....	15
Creating an image.....	16
Restoring an image.....	17
CLONE partitions or whole disks.....	18
Selecting the object.....	18
Performing the CLONE operation.....	19
COPY or MOVE partitions.....	21

This document offers some guidance in using the DFSee product to achieve the best results possible with the least effort, and avoid some common pitfalls.

Using the DFSee bootable CDROM

No doubt using DFSee from a bootable CDROM solves a few problems in case of disk problems. Using it however, also introduces a few potential problems:

1. Since DFSee is started **from the CDROM** the current directory is read-only

This is a problem for actions that want to write files into this current directory like:

- Open a logfile
- Find (lost) partitions

The easiest way to get around the problem is to change the current directory to a location where files **can** be written. This can be done from the DFSee commandline using the **CD** command, or from the menu using:

- File -> Change current directory

Note that other things like creating imagefiles or exporting the sectorlist will present a File-SaveAs dialog by default, so the current directory is of less concern. It is important of course, to select a directory that can be written to.

2. The bootable CDROM image emulates a diskette in drive **A:**
Because of this, the first or only diskette drive is accessible as **B:**, or not at all if two physical diskette drives are present. Note that the **A:** driveletter that will appear in the file-dialogs really represents the CDROM bootimage and can **NOT** be used to save files!

Another problem related to this *diskette emulation mode* is that it might not work on some (SCSI) systems because of subtle differences in the implementation of the El-Torrito standard used. When that happens, the symptoms might be anything from a hang during booting to simply booting from the harddisk ...

Do note that for the floppy-emulation to work, diskette-support **MUST** be enabled in the BIOS. So if you have a laptop without a diskette, do **NOT** turn that BIOS support off or the CDROM will not boot either!



3. The CDROM is using FreeDOS and the DFSDOS executable

While all DFSee versions share the same set of commands and menu options, and in general can perform the same actions, using a DOS environment does place some restrictions:

- Only FAT (or FAT32 with FreeDOS or DOS-7) are available for output files.
This means all logfiles, imagefiles and other files to be written by or accessed by DFSee need to reside on a FAT filesystem.

- DFSee will use the BIOS to access disks, so might be limited in some ways
The most common limitation is that large disks with more than 1024 cylinders can only be accessed completely if the BIOS has working INT13 extensions. Another is that SCSI, USB, Firewire and other non-standard disks are probably not accessible unless they have INT13 support built in to their disk-controllers.

INT13 is a standard for low-level disk-services available at PC-startup time (and later when running DOS). They can reside in the system-BIOS, an additional SCSI on-board BIOS, an on-board BIOS of other disk controllers or in software modules (disk-drivers) that load as part of the operating system.

The original implementation of INT13 services have been limited to the first 1024 cylinders of disk space (typically 7.8 GiB), but later implementations called 'Extended INT13' have lifted that limitation.

- The current FreeDOS version still seems to have some problems accessing files deeply nested in directory structures. This is probably not much of a problem for most DFSee usage, but it might explain some strange behavior like not showing any directory contents in deeply nested structures while browsing with a file-dialog.

The DFSee menu system and how to use it

Most users of computer systems these days are used to graphical user interfaces as used on Windows, MAC or even more recent Linux systems. These interfaces are controlled using windows on your screen, a menu-system to select actions to be performed, a mouse to point at objects on that screen and the keyboard to enter text or commands. This offers very good feedback and reduces the chances of making mistakes.

The other extreme, the classic interface used before the GUI's is the command line interface, with the screen displaying text output by the program and the keyboard used to enter commands. This requires the users to know or learn quite a lot about the program to be able to use all available functionality.



Since it is very hard to use a GUI program from a minimal system like a boot-diskette, DFSee is not using any GUI at all, instead it uses a compromise, where windows, menus, help-screens and dialogs are used but the interface is not graphical, it is still implemented as TEXT-mode.

It its current version **DFSee does NOT support or use a mouse** to navigate its user interface. When running under a GUI (text-window) like OS/2 or Windows you CAN use some mouse functionality like the clipboard however. Future versions of DFSee might support the mouse for navigation as well, and even real GUI-versions for some platforms are considered.

Navigating the DFSee menu

The following keys perform the basic navigation in the menu:

Enter	when just menu-headings are visible, will open the highlighted pulldown menu
Enter	when a pulldown is open, will activate or execute the highlighted menu item
Esc	when just menu-headings are visible, will close the menu and activate the commandline
Esc	when a pulldown is open, will close the pulldown
F10	when the menu is active, will close the menu and activate the commandline
F10	when the commandline is active, will activate the menu and open the default pulldown
Right arrow	when a submenu-item is highlighted in a pulldown, will open that submenu
Right arrow	when no submenu-item is highlighted in a pulldown, will move to the next pulldown
Left arrow	when a submenu is opened, will close that submenu
Left arrow	when no submenu is opened, will move to the previous pulldown
Down arrow	will move the highlight to the previous menu-item in the pulldown, and wrap around
Up arrow	will move the highlight to the next menu-item in the pulldown, and wrap around at the top

When only the menu-headings are visible, the first letter of each heading can be used as a quick-select key to open that menu-heading. Almost any other key used will open the highlighted heading. When the commandline is active the command ***menu x*** will activate the menu and open the heading with quick-select letter *x*.

When a pulldown menu is open, the letters that are highlighted (yellow) can be used as a quick-select key to activate (execute) that specific menu-item.

Context sensitive help and status bar description

Since menu-headings and item text can be rather cryptic, additional information is supplied:

1. The status bar, at the bottom of the screen, will display a full line of additional description about the menu-heading or menu-item currently selected. This is updated while you are scrolling through the menu.
2. At any time you can use the **F1** function-key to get help on the currently selected heading or menu-item. This will present you with a help-window dedicated to the task at hand.

Changing the default menu behaviour

The default behavior of the menu will cause the menu to be present, but without any pull-down menu opened after startup and after completing each command.

This ensures that the output from the command being executed is visible, while having the menu available to select the next action.

The menu behavior can be adapted to personal taste using switches on DFSee startup:

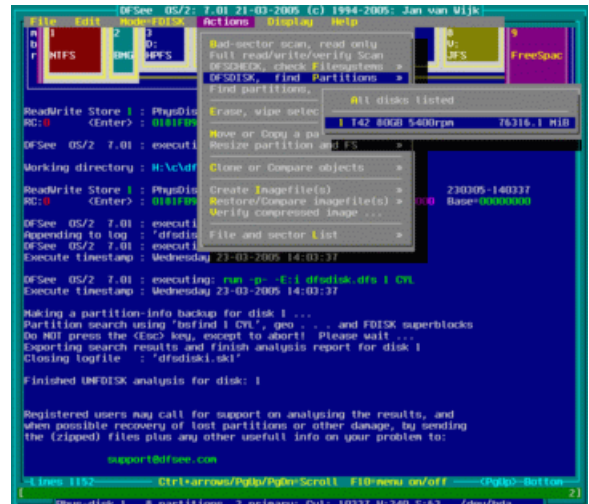
- | | |
|----------------------|--|
| DFSxxx -menu- | Do not activate the menu on startup, and do not automatically re-activate it after executing a menu selection |
| DFSxxx -M:1 | Do not open submenu when using right-arrow , instead the Enter key is required to open the submenu |
| DFSxxx -M:2 | Do not automatically open pulldown menus , instead an explicit key-press is required to open the pulldown |
| DFSxxx -M:3 | Combination of -M:1 and -M:2 |

In the above **DFSxxx** stands for any of of the available DFSee executables for the DOS, WINdows, Linux or OS2 platform.

DFSee DFSDISK procedures

The nightmare for every PC user, if you find yourself looking at:

- Blinking cursor on startup, PC does not boot
- Scaring messages like:
 - Error reading disk
 - Operating system not found
 - Invalid drive specification
 - Invalid partition table
 - Sector not found
- Or simply partitions missing after a mistake in FDISK or other tool ?



Most often you can **recover lost partitions** using DFSee!

The DFSDISK functionality in DFSee is based on an extensive search of the disk(s) for any remaining partition-tables, bootsectors or LVM related sectors. The results of this search is stored in several files for every disk examined (6 of them with DFSee 10.x).

These files (DFSDISKI.*) are the basis for an analysis that may result in the cration of a recovery script (.DFS) that will recreate the missing or damaged partitions, and/or fix any other problems found.

Because of the complexity of the matter, and the many variations is disk layout and filesystems, the analysis is NOT automatic and requires someone knowledgable about disks, partitions and the problems that may occur with them. It is usually done by FSYS-software *SUPPORT*, and in that case does **require** a valid registration.

Getting the DFSDISKI.* files

To create the required analysis files you will use the *dfsdproc.dfs* script that comes with the standard DFSee distributions including the CDROM. This script requires that the **current directory** when running this script is writable, and has enough space for the files (typically e few hundred Kb per analyzed disk).

There are two ways to run the DFSee DFSDISK procedure:

DFSDISK from the menu inside DFSee

To collect the required files for one or all disks use the following menu selections:

- Actions -> DFSDISK, find Partitions -> ... select one or ALL disks

Then fill in the dialog using mostly the defaults, except for the base filename to use.

In case nothing was found due to geometry problems, you could instruct DFSee to search ALL sectors instead of just the ones on cylinder boundaries, in that same dialog.

DFSDISK from the operating system command line

You would need the DFSDPROC.DFS script that comes with DFSee, plus:

- For DOS or Win9x bootdiskettes DFSDISK.BAT and DFSDOS.EXE.
- For Win-NT/2000/XP DFSDISK.BAT and DFSWIN.EXE
- For OS/2 DFSDISK.CMD and DFSOS2.EXE
- For Linux the 'dfsdisk' script and the 'dfsee' executable

You simply run the script-file (.BAT or .CMD) without any parameters and it will collect SIX files for every physical disk you have. (DFSDISK?.*). By default it will handle all disks, but you can specify parameters to do just a single disk, or search in ALL sectors on a disk. Use *dfsdisk -?* to get usage information

Analysis of the results

Based on the DFSDISK?.* information a recovery script can be created that can simply be run from the operating system command line like:

DFSxxx.exe run recover.dfs

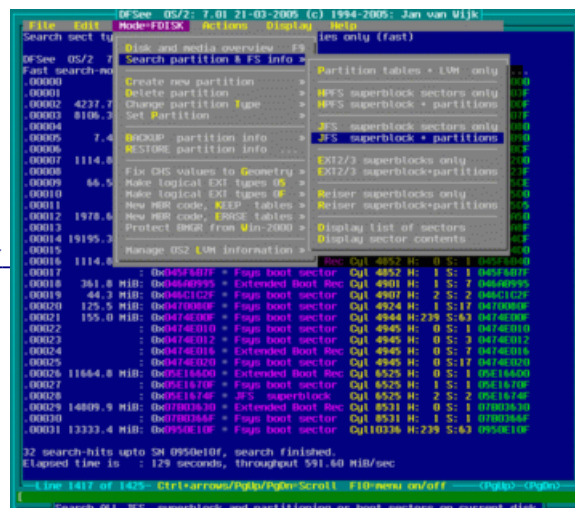
You will most likely need assistance to do analyse the information, and that DOES require you to have [or buy](#) a registration!

If you have the wanted information collected in the files, you can send them to DFSee SUPPORT at:

support@dfsee.com

To save space, I advice to compress all the files before sending them, for example to a single DFSDISKI.ZIP

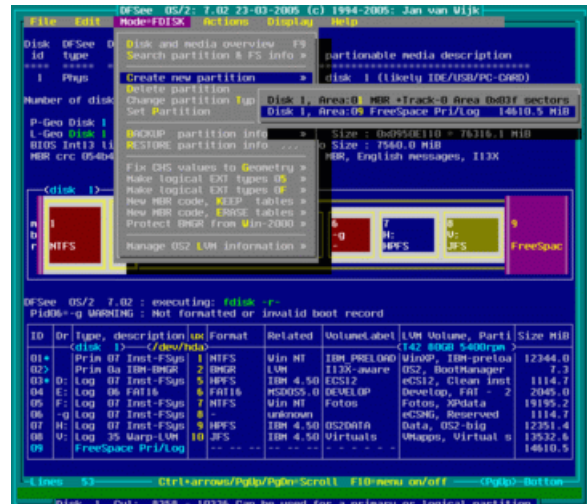
If you want to attempt the analysis yourself, or simply want to learn more about the matter you can also check the [DFSee DFSDISK descriptions and usage](#)



CREATE new partitions in freespace areas

You can create new partitions from the DFSee commandline using the CR command, see [DFSee COMMAND overview](#), or from the menu as shown above.

When using the menu, the CREATE selection opens a submenu with all available freespace areas.



Only freespace areas that are large enough, and in a proper location to contain new partitions will be selectable. Freespace areas might not be selectable when:

- They are designated 'wasted' because they are outside an existing extended container, and/or no more primaries can be created. The limit is FOUR primaries, or one extended container plus THREE primaries.
A primary partition can NEVER be between two logical partitions, all logicals need to be within a single area called the extended container.
- The freespace area is too small to contain a minimal size cylinder-aligned partition

Select the one you want to use for the new partition, and on ENTER you will be presented with a CREATE specific dialog that allows you to specify all relevant information.

The CREATE dialog allows you to specify:

- The kind of partition, which can be either of:
 - Logical partition
 - Primary partition

If creating either type is not possible, the option will not be shown ...

- Location of the new partition, either of:
 - At start freespace
 - At end freespace

If you want to place the partition somewhere in the MIDDLE of a freespace area, you will have to use the CR command manually, and use the '-at:' option with a megabyte or cylinder offset.

- The size in megabytes for the new partition
This will be rounded up to make the new partition align to cylinder boundaries. In practice this may increase the specified size with up to 7 megabytes.
 - The partition type
The most common types can be selected directly from the selection list, and for more obscure values you can use the 'Specify custom type value (HEX)' selection to specify any value between 0x01 and 0xff.
 - Include LVM info
This option causes the LVM dialog to be activated after the partition has been created, so you can specify the required (OS2 type) LVM information for this partition/volume in one go.
 - Create NEW mode, clear existing bootsector and old LVM info
This option will instruct the CREATE command to CLEAR the bootsector for the newly created partition with a 0xF6 pattern, just like regular FDISK programs do. This avoids problems when you want to format the new partition and the FORMAT program might pick up 'old' size values from the bootsector instead of using the new partition size.
- For recovery purposes however, you do NOT want the bootsector to be cleared, so use:
- Recovery mode, keep existing bootsector and LVM info intact

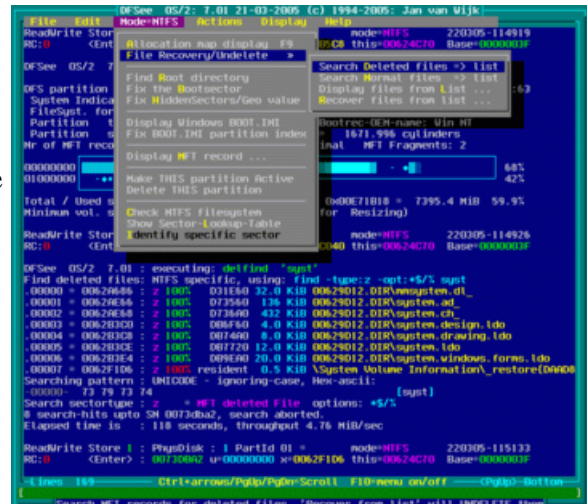
After completing the CREATE dialog, you will get an additional confirmation dialog with the new partition details displayed on the screen so you can verify them.



UNDELETE / RECOVER files from HPFS, NTFS, FAT, JFS

Find your deleted file(s) on an HPFS/NTFS partition, or find regular files on such a partition that has become inaccessible for the operating system (no driveletter assigned, not 'mounted'), or that has been formatted by accident.

Recover the file-data for all or selected files to another volume (driveletter).



Undelete or file-recovery in DFSee is a three to four step process:

1. Select the partition in question

For **UNDELETE** this must be an HPFS, JFS or NTFS partition, for recovery of normal files it can also be a FAT(32) partition. Select the partition using:

- File -> Open partition to work with

2. Find deleted or normal files, collecting found ones in a list. Use one of:

- Mode=**xxx** -> File Recovery/Undelete -> Search deleted files => list
- Mode=**xxx** -> File Recovery/Undelete -> Search normal files => list

Depending on the current filesystem **xxx** you can specify part of the name for the file(s) to be found in the next dialog.

While searching, a reference will be added to the DFSee sector-list for every file found that matches this partial filename and the full path+filename for the file will be displayed together with a recovery prognosis.

Note that searching for files on a large disk, may take a very long time. Expect between less than 1 to more than 10 minutes per gigabyte, depending on the speed of your harddisk, the filesystem used and the amount of freespace.

3. Display the list, using a selection wildcard specification (**optional**):

- Mode=**xxx** -> File Recovery/Undelete -> Display files from list ...

This optional step might be usefull to find out what the best wildcard is to get exactly the file(s) you need to recover. The wildcard may describe any part of the full path+filename displayed while searching, and it can contain multiple wildcard-characters:

- * representing one or more characters in the path+filename
- ? representing exactly one character in the path+filename

As an example: **mydocs*test?.doc*

would display/recover all **.DOC** files with a name starting with **test** plus just one character that have **mydocs** somewhere in the directory path.

4. Recover the file-data for all or selected files to another volume (driveletter).

- Mode=**xxx** -> File Recovery/Undelete -> Recover files from list ...

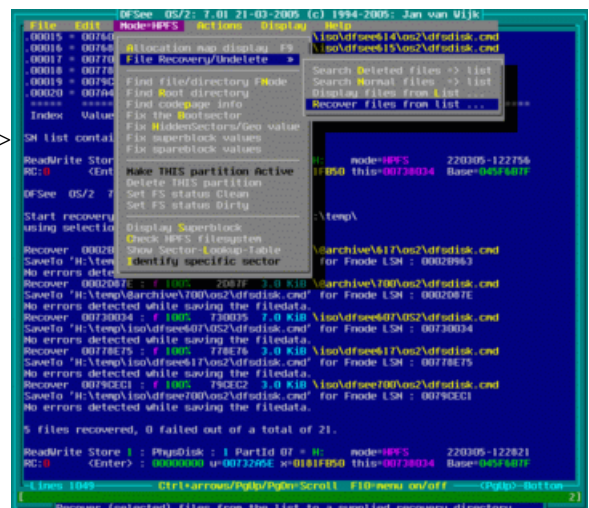
This will first present a dialog where you need to specify the directory where the filedata will be recovered to. Of course this must be a writable location with enough freespace to hold the filedata for all files to be recovered. In this destination directory, the files will be recreated with their full original path and filename when available, and the data for the file is copied over.

Note: For HPFS each component in the path and the filename will be limited to a maximum of 15 characters due to the limited-length names as stored in the file and directory FNODE structures. This limitation might be lifted in future DFSee releases.

After specifying the destination directory, the next dialog allows you to specify a selection wildcard, exactly as with the optional display step described above.

After this the files will be recovered one by one, with progress information displayed.

For later reference and checking the results, it is advised to start a logfile before starting the recovery procedure.



IMAGING using RAW or compressed files

Create an imagefile from the currently selected object (disk/partition) or restore an imagefile.

Imagefiles are (binary) files that hold a complete representation of a disk-partition or even a complete disk. Their use is in backup, system recovery and moving contents from one system to another. In DFSee there are two main types of imagefiles:



1. RAW

These are one-to-one exact sector copies of the original partition or disk in a binary file. The data is NOT compressed or changed in any way and nothing is added to the files for identification or other purposes.

These are the kind of images that may be shared with other applications since it is a defacto standard. It is used a lot with diskette images, but also by Virtual-machine implementations like VPC or SVISTA as far as uncompressed disk images are used to represent the hard-disk of the virtual PC.

2. Compressed

The main difference with the RAW format is that they are well, *compressed* meaning smaller ... DFSee uses a mixture of LZW and RLE compression methods to achieve maximum compression ratio and imaging speed for 'typical' situations. In addition it may use *SmartUse* filesystem information to skip unused areas of partitions when creating or restoring imagefiles. This results in tremendous savings in filesize and processing-time for filesystems that have lots of unused (free) space.

Currently *SmartUse* imaging is implemented for **HPFS**, **NTFS**, **JFS**, **EXT2/3**, **Reiser** and **FAT** filesystems.

There are three other properties that are useful with *Compressed* images mainly, but do work for RAW as well:

- **Info header**

This adds a DFSee specific ASCII header to the imagefiles, they can be very useful to identify the imagefile in question. You can view this info quite easily using the standard 'TYPE filename' command on the operating system command line or a 'view' option as offered by many filemanagers. Adding this header can be suppressed using the **-Raw** option on the SIM command or checking the corresponding checkbox in the dialog.

Note that DFSee compressed images are NOT ZIP-files! First of all, they use a slightly different compression method, and second it is not a 'file-archive' with a directory like regular ZIP-files are.

- **Size limit**

This limits the maximum size of the imagefile(s) being created to the specified size (or 2 GiB when not specified). This is useful to create files that will fit on a diskette, a CDROM or DVD or that are compatible with the maximum filesize for the filesystem used to store the imagefiles.

- **Media Change**

This informs the DFSee imaging process that the files will be written to removable media (like diskette or CDROM) and that it should prompt the user to change the media involved before starting each imagefile.

Note that to write to multiple CDROMs using DFSee, your CD-writer software must support *Streaming* write mode, where the CDROM gets a driveletter assigned that can be written to just like any other drive.

RSJ for OS/2 or Windows is such a product, and for Windows only you can use Adaptec Easy CD creator or Roxio DirectCD (which should allow writing to DVD's too).

Selecting the object

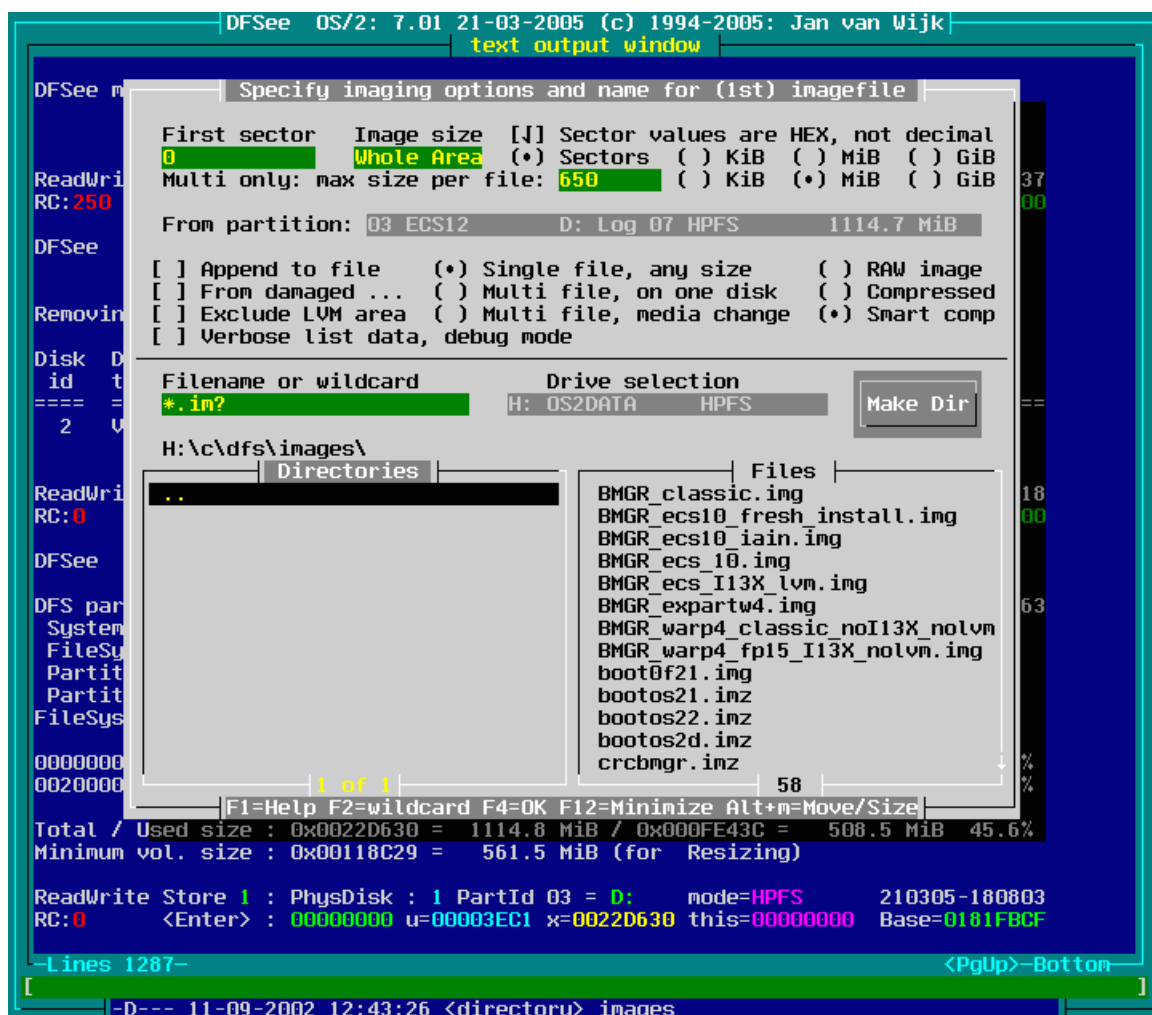
Although you can use the "File -> Open object to work with" menu to open an object and then create an image "from Current object", it is much more convenient to use the available menu selections in the imaging submenu. That way all selections needed to create the image are made from that single menu-selection plus a dialog:

- Actions -> Create Imagefile(s) -> from a Disk ...
- Actions -> Create Imagefile(s) -> from a Partition ...
- Actions -> Create Imagefile(s) -> from a Volume (diskette) ...

For diskette-images you would use a Volume, in most other situations you would use a Partition and in some rare cases you might want images for a complete disk.

Creating an image

You can create images from the DFSee commandline using the IMAGE command, see [DFSee COMMAND overview](#), or from the menu as shown above. When using the menu, or when specifying an incomplete IMAGE command, you will be presented with an imaging specific dialog that allows you to specify all relevant information.



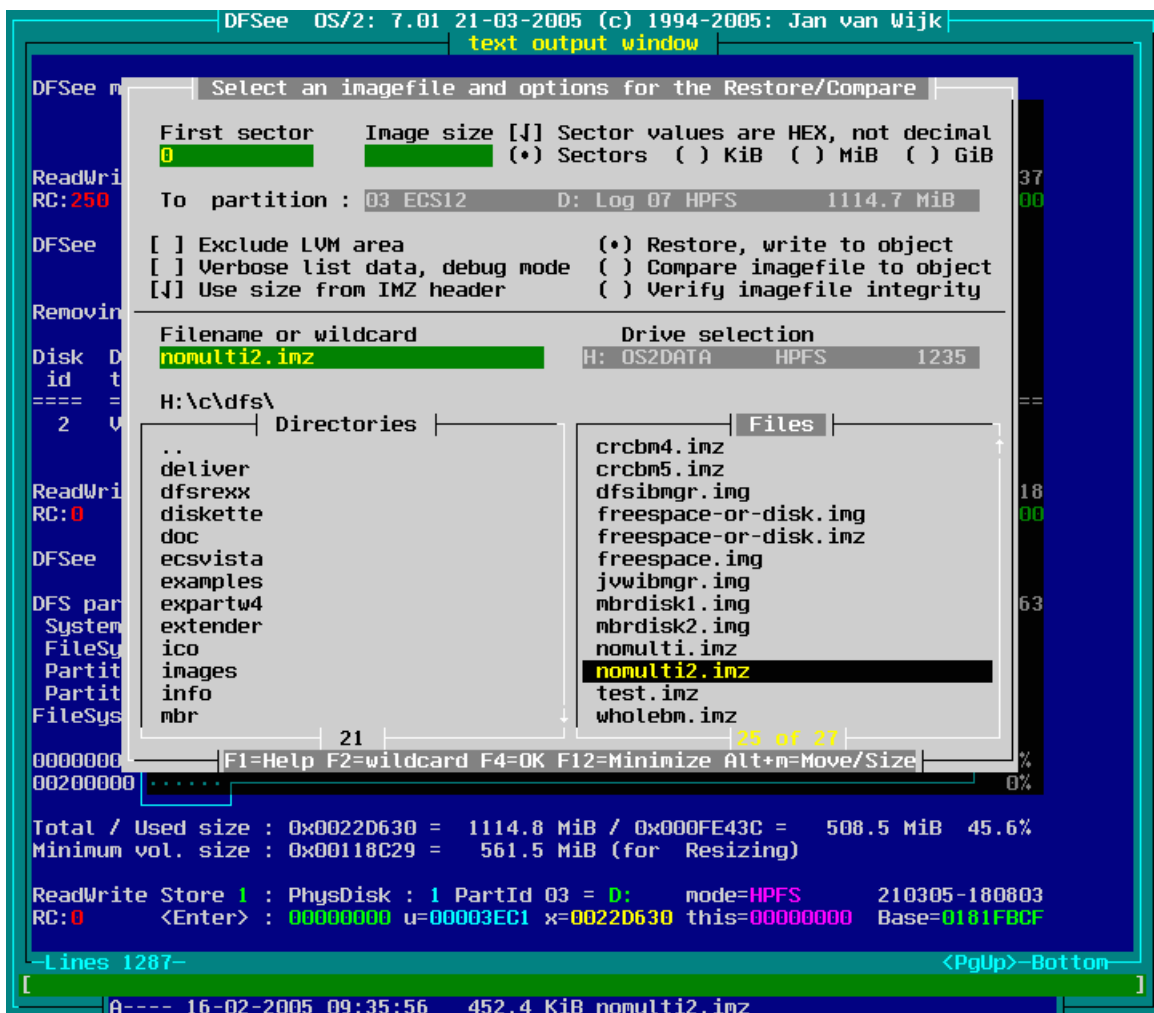
Restoring an image

You can create images from the DFSee commandline using the RESTORE command, see [DFSee COMMAND overview](#), or from the menu using:

- Actions -> Restore/Compare Imagefile(s) ...

This will present you with a combined File-Open and options dialog where you can select an imagefile to be restored as well as any specific options.

The restore logic will automatically handle compression, multiple files *SmartUse* and media-change where needed.



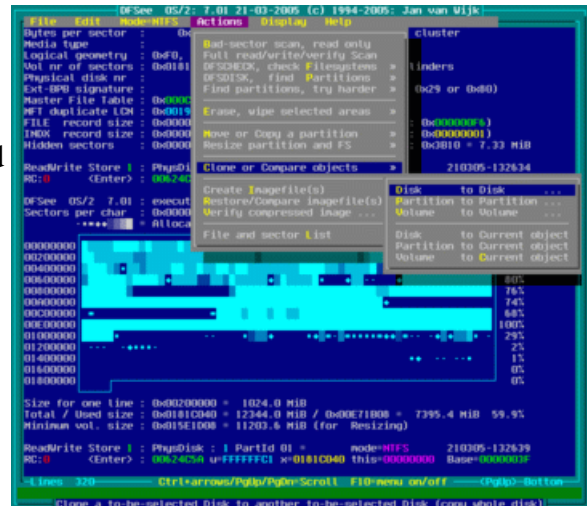
CLONE partitions or whole disks

Copy the contents of another disk/partition to the currently selected object (disk/partition).

This will result in an exact sector-by-sector copy called a CLONE.

Selecting the object

You can use either the "File -> Open object to work with" menu to open an object and then clone from another object "to Current object", or use the more specific menu selections in the imaging submenu.



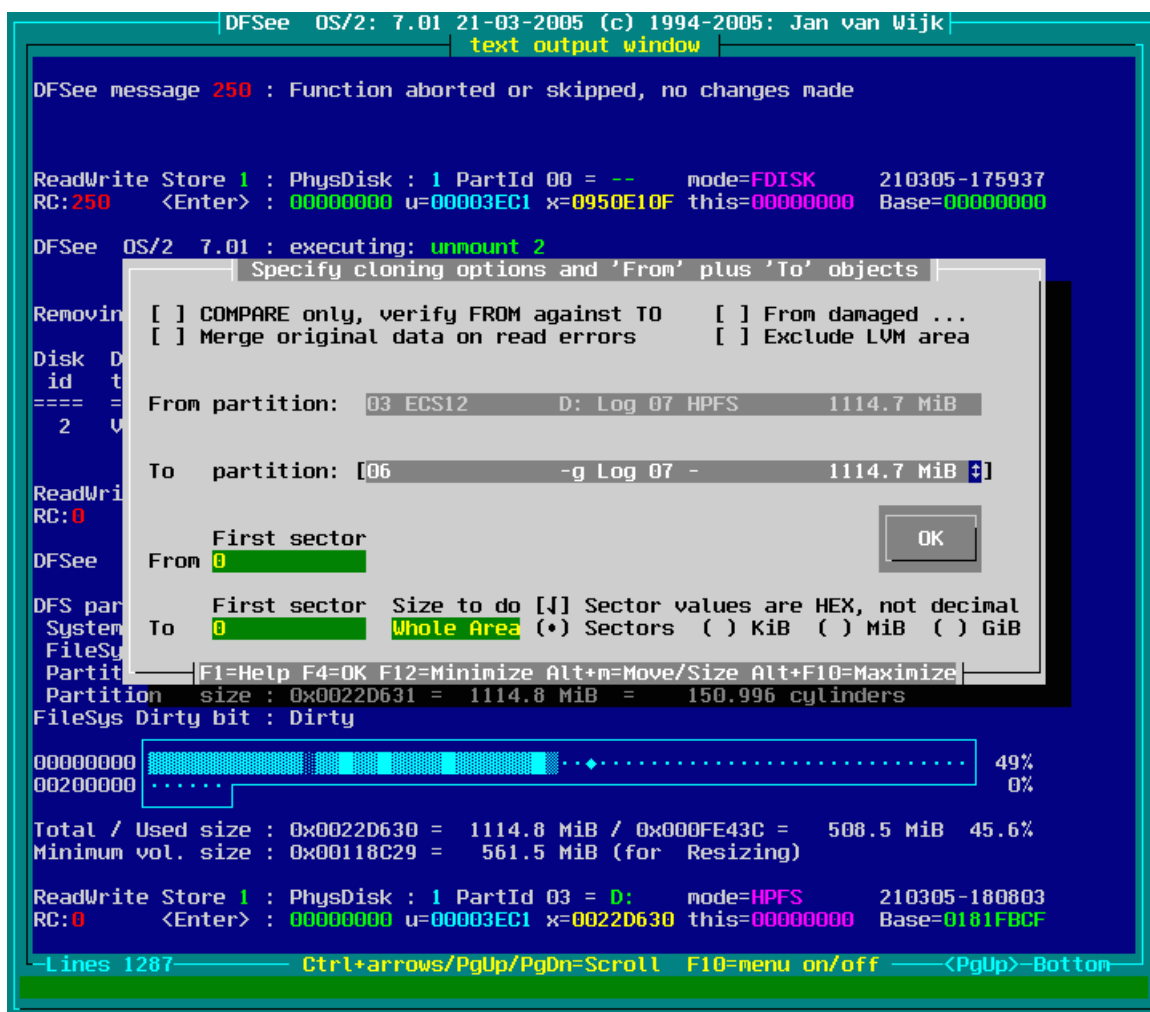
Either way, most selections needed are made from that single menu-selection plus a dialog:

- Actions -> Clone or Compare objects -> Disk to Disk
This would be useful to make a copy of an existing disk to a new, larger one without having to install everything again on the new disk. Note that any partitions will be exact clones, NO resizing takes place. This means the new disk, when larger, will leave you with a freespace area at the end of the disk that you can use to create new partitions, or to use with resizing and moving the existing partitions.
Another purpose would be cloning a DAMAGED disk (bad sectors) to try and get as much data off it as possible. To do this, make sure you check the "Some damaged" option in the dialog, to optimize the cloning process for this purpose.
- Actions -> Clone or Compare objects -> Partition to Partition
This is very useful to make a quick backup copy of partition contents to an identical sized one, preferably on another disk. Making a backup copy of your boot-partition(s) in a fully installed and configured state, but clean of any garbage and viruses, allows a very quick recovery in case a virus or other disaster hits.
Simply use the clone "partition to Partition" again to copy the partition contents back ...
- Actions -> Clone or Compare objects -> Volume to Volume
- Actions -> Clone or Compare objects -> Disk to Current object
- Actions -> Clone or Compare objects -> Partition to Current object
- Actions -> Clone or Compare objects -> Volume to Current object

Performing the CLONE operation

You can clone objects from the DFSee commandline using the CLONE command, see [DFSee COMMAND overview](#), or from the menu as shown above.

When using the menu, or when specifying an incomplete CLONE command, you will be presented with a cloning specific dialog that allows you to specify all relevant information.



Some of the options available are:

- **COMPARE only, verify FROM against TO**
This will COMPARE the sectors of the TO and FROM objects instead of copying them, to allow a verification of a cloning operation, or simply to see if two objects have exactly the same contents.
- **Merge original data on read errors**
This will leave the original data on the destination object when read-errors happen during cloning instead of filling them with a fixed 0xFE pattern.
This can be useful to clone damaged disks that have random failures in a few passes, or to 'merge' a freshly formatted filesystem with a damaged one containing data.
- **From damaged ...**
This will reduce the buffersize during cloning to 1 sector, making the operation MUCH slower but also reducing the number of sectors affected by any read errors (bad sectors).
- **Exclude LVM area**
This will exclude the area at the end of (OS/2 style) LVM partitions that holds the bad-block relocation and drive-linking information. This can be useful in recovery situations, since after cloning that information will be invalid and needs to be recreated anyway.
- **First sector**
You can specify to start the clone at a specific sector instead of the start of the object. Since you can specify the start for the TO and FROM object separately, you can use this to MOVE data within a partition or relative to the start of the object.

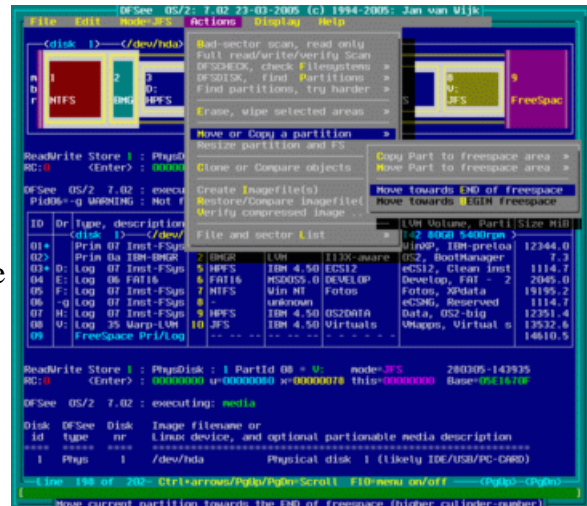
Note that the CLONE operation supports overlapping TO and FROM areas, so it is possible to move data WITHIN a partition or other object.
- **Size to do**
This can limit the size to be copied to a value less than the whole object.

COPY or MOVE partitions

Moving or copying a partition deals with the partition contents as well as the partition information (partition-tables and LVM), and is the easiest way to do this. It 'adds' functionality to the basic CLONE command.

You can MOVE or COPY partitions from the DFSee command line using the MOVE command (and -c option), see [DFSee COMMAND overview](#), or from the menu using one of following selections:

- Actions -> Move or Copy a partition -> Copy Part to freespace area
Create a copy of the currently selected partition in a freespace area to be selected from a submenu-list. Only freespace areas that are large enough to contain the copy will be selectable. The size of the partition will stay exactly the same, there is NO automatic resize. When applicable, LVM information will be created that is derived from the old partition, with post-fixes to make the names and IDs unique.
- Actions -> Move or Copy a partition -> Move Part to freespace area
Create a copy of the currently selected partition in a freespace area to be selected from a submenu-list, and delete the original one. Only freespace areas that are large enough to contain the partition will be selectable. The size of the partition will stay exactly the same, there is NO automatic resize. When applicable, LVM information will be moved with the partition.
- Actions -> Move or Copy a partition -> Move towards END of freespace
This will MOVE the currently partition towards the end of the freespace area that is directly after the partition. You can specify how far the partition should be moved, with a maximum of the freespace size. The result will be that some or all of the freespace will now be IN FRONT of the partition.
- Actions -> Move or Copy a partition -> Move towards BEGIN freespace
This will MOVE the currently partition towards the begin of the freespace area that is directly before the partition. You can specify how far the partition should be moved, with a maximum of the freespace size. The result will be that some or all of the freespace will now be AFTER of the partition.



The actual copying or moving of the partition data will be done using a CLONE command, and the required partitioning and LVM commands will be determined and executed where needed.