NAME

cp, mv, rm, ln, cpdir, clone − copy, move, remove, link

SYNOPSIS

cp [**−pifsmrRvx**] *file1 file2*
cp [**−pifsrRvx**] *file ... dir*

mv [**−ifsmvx**] *file1 file2*
mv [**−ifsvx**] *file ... dir*

rm [**−ifrRvx**] *file ...*

ln [**−ifsSmrRvx**] *file1 file2*
ln [**−ifsSrRvx**] *file ... dir*

cpdir [**−ifvx**] *file1 file2*

clone [**−ifsSvx**] *file1 file2*

DESCRIPTION

The utilities **cp**, **mv**, **rm**, and **ln** do basic file management: copying, renaming or moving, deletion, and creating links. (The **cpdir** and **clone** utilities are easy to use aliases for copying or linking whole trees. They are the same as **cp −psmr** and **ln −fmr**)

The first synopsis form of the utilities **cp**, **mv**, and **ln** is used if only two arguments are given, and the second argument is not a directory. The source and target file are then the two files given.

If the second synopsis form is used then the last argument must be a directory. Each of the files is copied, moved or linked into this directory.

A file is by default copied by **cp** without looking at its type, so symlinks are followed and devices are opened and read from or written to. Links between files are ignored. This behavior can be changed by using the proper options.

The **mv** utility uses the **rename**(2) call to rename or move files. If source and target are on different devices however, then **mv** will use **cp −pr** to copy the files or directory trees.

Each utility continues with the next file on errors, except on I/O errors.

OPTIONS

−p Copy the file attributes like mode, owner, group and time of last modification. Normally only the mode is copied to a new file with the file creation mask applied. Setuid bits are cleared if setting the ownership fails.

−i Ask if ok to overwrite, replace or remove. **Mv** and **rm** will ask this automatically if interactive and the target file is writable. **Cp** will fail if the target cannot be written, **ln** will always fail if the target exists.

−f Makes **cp** remove a target file before copying if it is not writable, **mv** removes an existing target without asking, **rm** does not report any errors, and **ln** removes an existing target file before linking. The last of **−i** and **−f** wins for **mv** if both flags are set, the other utilities do something sensible, like asking before forcefully removing.

−s Make a symlink instead of a normal link. For utilities other than **ln** this flag means "copy similar". The modified time is always copied for **cp −s** and the other attributes are copied if a new file is created. The normal POSIX required patronizing like applying the file creation mask or clearing setuid bits is not done.

−S Make a symlink if a normal link cannot be made because source and target are on different devices. The symlink is required to really refer back to the source, meaning that a/b must exist in the call **ln −S a/b c/d**, and that the symlink from c/d must lead back to a/b. So the symlink will be created as if **ln −s ../a/b c/d** was called. If the target is a full path, but the source is not then an error will be given saying that this is "too difficult."

−m Merge trees. The first synopsis form is assumed, and the files from one tree are merged into the other. There is no "if it’s a directory the put it into that directory" trickery here.

−r, **−R** Recursively copy, remove, or link. If the source is a directory then the files in this directory are copied to similarly named files in the target directory. Special files are copied as new special files, they are not read or written. Symlinks are still expanded and the link structure ignored with **−R**. The **−r** flag does copy symlinks as symlinks and keeps the link structure intact. (Note that **−R** is invented by POSIX as a replacement for the classic **−r** option of older copy commands that did read special files. The standard says that **−r** is implementation defined, so that's why this flag is better than **−R** in this implementation of **cp**.) For **rm** and **ln** both flags mean the same. **Ln** will recursively link the files in the trees, except symlinks, they are copied. If symlinks are created with **ln −rs** or **ln −rS** then they are required "to work" as described with the **−S** flag.

−v Verbose. Show what is done on standard output.

−x Do not cross mount points. Empty directories will be created if the source directory is a mount point on a copy, move or link. A mount point will not be removed or traversed recursively. This flag allows one to copy the root device, e.g. **cpdir −x / /mnt**.

SEE ALSO

cat(1), **mkdir**(1), **rmdir**(1), **mkdir**(2), **rmdir**(2), **link**(2), **unlink**(2), **rename**(2), **open**(2), **read**(2), **write**(2), **opendir**(3).

NOTES

All the utilities described are links to the same program.

BUGS

Mv should first copy a tree across devices and then remove the source tree if there was no error. Instead, each file in the tree is copied and immediately removed. On error you may be left with two half-filled trees, together containing all of the files. You may have to restart the move with **mv −m**.

Rm should be able to remove arbitrarily deep trees.

AUTHOR

Kees J. Bot (kjb@cs.vu.nl)