NAME

rm, **unlink** — remove directory entries

SYNOPSIS

rm [**-f** | **-i**] [**-dIPRrvWx**] *file ...*
unlink [**--**] *file*

DESCRIPTION

The **rm** utility attempts to remove the non-directory type files specified on the command line. If the permissions of the file do not permit writing, and the standard input device is a terminal, the user is prompted (on the standard error output) for confirmation.

The options are as follows:

-d Attempt to remove directories as well as other types of files.

-f Attempt to remove the files without prompting for confirmation, regardless of the file's permissions. If the file does not exist, do not display a diagnostic message or modify the exit status to reflect an error. The **-f** option overrides any previous **-i** options.

-i Request confirmation before attempting to remove each file, regardless of the file's permissions, or whether or not the standard input device is a terminal. The **-i** option overrides any previous **-f** options.

-I Request confirmation once if more than three files are being removed or if a directory is being recursively removed. This is a far less intrusive option than **-i** yet provides almost the same level of protection against mistakes.

-P Overwrite regular files before deleting them. Files are overwritten three times, first with the byte pattern 0xff, then 0x00, and then 0xff again, before they are deleted. Files with multiple links will not be overwritten nor deleted and a warning will be issued. If the **-f** option is specified, files with multiple links will also be overwritten and deleted. No warning will be issued.

Specifying this flag for a read only file will cause **rm** to generate an error message and exit. The file will not be removed or overwritten.

N.B.: The **-P** flag is not considered a security feature (see **BUGS**) .

-R Attempt to remove the file hierarchy rooted in each *file* argument. The **-R** option implies the **-d** option. If the **-i** option is specified, the user is prompted for confirmation before each directory's contents are processed (as well as before the attempt is made to remove the directory). If the user does not respond affirmatively, the file hierarchy rooted in that directory is skipped.

-r Equivalent to **-R**.

-v Be verbose when deleting files, showing them as they are removed.

-W Attempt to undelete the named files. Currently, this option can only be used to recover files covered by whiteouts in a union file system (see `undelete`(2)).

-x When removing a hierarchy, do not cross mount points.

The **rm** utility removes symbolic links, not the files referenced by the links.

It is an error to attempt to remove the files `/`, **.** or **..** .

When the utility is called as **unlink**, only one argument, which must not be a directory, may be supplied. No options may be supplied in this simple mode of operation, which performs an `unlink`(2) operation on the passed argument. However, the usual option-end delimiter, **--** , may optionally precede the argument.

EXIT STATUS

The **rm** utility exits 0 if all of the named files or file hierarchies were removed, or if the **-f** option was specified and all of the existing files or file hierarchies were removed. If an error occurs, **rm** exits with a value >0.

NOTES

The **rm** command uses `getopt`(3) to parse its arguments, which allows it to accept the '`--`' option which will cause it to stop processing flag options at that point. This will allow the removal of file names that begin with a dash ('-'). For example:

```
rm -- -filename
```

The same behavior can be obtained by using an absolute or relative path reference. For example:

```
rm /home/user/-filename
rm ./-filename
```

When **-P** is specified with **-f** the file will be overwritten and removed even if it has hard links.

EXAMPLES

Recursively remove all files contained within the `foobar` directory hierarchy:

```
$ rm -rf foobar
```

Any of these commands will remove the file `-f`:

```
$ rm -- -f
$ rm ./-f
$ unlink -f
```

COMPATIBILITY

The **rm** utility differs from historical implementations in that the **-f** option only masks attempts to remove non-existent files instead of masking a large variety of errors. The **-v** option is non-standard and its use in scripts is not recommended.

Also, historical BSD implementations prompted on the standard output, not the standard error output.

SEE ALSO

`chflags`(1), `rmdir`(1), `undelete`(2), `unlink`(2), `fts`(3), `getopt`(3), `symlink`(7)

STANDARDS

The **rm** command conforms to .

The simplified **unlink** command conforms to Version 2 of the Single UNIX Specification ("SUSv2").

HISTORY

A **rm** command appeared in Version 1 AT&T UNIX.

BUGS

The **-P** option assumes that the underlying storage overwrites file blocks when data is written to an existing offset. Several factors including the file system and its backing store could defeat this assumption. This includes, but is not limited to file systems that use a Copy-On-Write strategy (e.g. ZFS or UFS when snapshots are being used), Flash media that are using a wear leveling algorithm, or when the backing datastore does journaling, etc. In addition, only regular files are overwritten, other types of files are not.