-- --

IEEE P1003.2 Draft 11.2 – September 1991

It is less clear that error messages regarding unlinkable files should be suppressed. Although this is historical practice, this standard does not permit the `−f` option to suppress such messages.

When given the `−r` and `−i` options, historical versions of `rm` prompt the user twice for each directory, once before removing its contents and once before actually attempting to delete the directory entry that names it. This allows the user to ‘‘prune’’ the file hierarchy walk. Historical versions of `rm` were inconsistent in that some did not do the former prompt for directories named on the command line and others had obscure prompting behavior when the `−i` option was specified and the permissions of the file did not permit writing. The POSIX.2 `rm` differs little from historic practice, but does require that prompts be consistent. Historical versions of `rm` were also inconsistent in that prompts were done to both standard output and standard error. POSIX.2 requires that prompts be done to standard error, for consistency with `cp` and `mv` and to allow existing extensions to `rm` that provide an option to list deleted files on standard output.

The `rm` utility is required to descend to arbitrary depths so that any file hierarchy may be deleted. This means, for example, that the `rm` utility cannot run out of file descriptors during its descent, i.e., if the number of file descriptors is limited, `rm` cannot be implemented in the historical fashion where a file descriptor is used per directory level. Also, `rm` is not permitted to fail because of path length restrictions, unless an operand specified by the user is longer than {PATH_MAX}.

4.54 `rmdir` — Remove directories

4.54.1 Synopsis

`rmdir [−p]` *dir* . . .

4.54.2 Description

The `rmdir` utility shall remove the directory entry specified by each *dir* operand, which shall refer to an empty directory.

Directories shall be processed in the order specified. If a directory and a subdirectory of that directory are specified in a single invocation of the `rmdir` utility, the subdirectory shall be specified before the parent directory so that the parent directory will be empty when the `rmdir` utility tries to remove it.

4.54.3 Options

The `rmdir` utility shall conform to the utility argument syntax guidelines described in 2.10.2.

The following option shall be supported by the implementation:

`−p` Remove all directories in a pathname. For each *dir* operand:

(1) The directory entry it names shall be removed.

(2) If the *dir* operand includes more than one pathname component, effects equivalent to the following command shall occur:

`rmdir −p $(dirname` *dir* `)`

4.54.4 Operands

The following operand shall be supported by the implementation:

dir A pathname of an empty directory to be removed.

4.54.5 External Influences

4.54.5.1 Standard Input

None.

4.54.5.2 Input Files

None.

4.54.5.3 Environment Variables

The following environment variables shall affect the execution of `rmdir`:

LANG This variable shall determine the locale to use for the locale categories when both **LC_ALL** and the corresponding environment variable (beginning with **LC_**) do not specify a locale. See 2.6.

LC_ALL This variable shall determine the locale to be used to override any values for locale categories specified by the settings of **LANG** or any environment variables beginning with **LC_**.

LC_CTYPE This variable shall determine the locale for the interpretation of sequences of bytes of text data as characters (e.g., single- versus multibyte characters in arguments).

LC_MESSAGES This variable shall determine the language in which messages should be written.

4.54.5.4 Asynchronous Events

Default.

4.54.6 External Effects

4.54.6.1 Standard Output

None.

4.54.6.2 Standard Error

Used only for diagnostic messages.

4.54.6.3 Output Files

None.

4.54.7 Extended Description

None.

4.54.8 Exit Status

The `rmdir` utility shall exit with one of the following values:

0 Each directory entry specified by a *dir* operand was removed successfully.

>0 An error occurred.

4.54.9 Consequences of Errors

Default.

4.54.10 Rationale. *(This subclause is not a part of P1003.2)*

Examples, Usage

On historical System V systems, the `−p` option also caused a message to be written to the standard output. The message indicated whether the whole path was removed or part of the path remains for some reason. The Standard Error subclause requires this diagnostic when the entire path specified by a *dir* operand is not removed, but does not allow the status message reporting success to be written as a diagnostic.

If a directory `a` in the current directory is empty except it contains a directory `b` and `a/b` is empty except it contains a directory `c`,

```
rmdir -p a/b/c
```

will remove all three directories.

The `rmdir` utility on System V also included an `−s` option that suppressed the informational message output by the `−p` option. This option has been omitted because the informational message is not specified by POSIX.2.

4.55 sed — Stream editor

4.55.1 Synopsis

`sed [−n]` *script* [*file* . . .]

`sed [−n] [−e` *script*] . . . [`−f` *script_file*] . . . [*file* . . .]

4.55.2 Description

The `sed` utility is a stream editor that shall read one or more text files, make editing changes according to a script of editing commands, and write the results to standard output. The script shall be obtained from either the *script* operand string or a combination of the option-arguments from the `−e` *script* and `−f` *script_file* options.

4.55.3 Options

The `sed` utility shall conform to the utility argument syntax guidelines described in 2.10.2, except that the order of presentation of the `−e` and `−f` options is significant.

The following options shall be supported by the implementation: