IEEE P1003.2a Draft 8 – December 1991

5.24 `renice` — Set system scheduling priorities of running processes

5.24.1 Synopsis

`renice` [`−n` *increment*] [`−g` | `−p` | `−u`] *ID* . . .

Obsolescent Versions :

`renice` *nice_value* [`−p`] *pid* . . . [`−g` *gid* . . .] [`−p` *pid* . . .] [`−u` *user* . . .]

`renice` *nice_value* `−g` *gid* . . . [`−g` *gid* . . .] [`−p` *pid* . . .] [`−u` *user* . . .]

`renice` *nice_value* `−u` *user* . . . [`−g` *gid* . . .] [`−p` *pid* . . .] [`−u` *user* . . .]

5.24.2 Description

The `renice` utility shall request that the system scheduling priorities (see 7
2.2.2.212) of one or more running processes be changed. When a process ID is 7
`renice`d (the default), the applicable processes are specified by their process IDs. 7
When a process group is `renice`d, the request applies to all processes in the pro- 7
cess group. 7

The system scheduling priority shall be bounded in an implementation-defined manner. If the requested *increment* (or *nice_value* in the obsolescent versions) would raise or lower the system scheduling priority of the executed utility beyond implementation-defined limits, then the limit whose value was exceeded shall be used.

When a user is `renice`d and {_POSIX_SAVED_IDS} (see 2.9.3 in POSIX.1 {8}) is
defined, the request applies to all processes whose saved set-user-ID matches the 7
user ID corresponding to the user. When a user is `renice`d and
{_POSIX_SAVED_IDS} is not defined, the request applies to all processes whose 7
real user ID matches the user ID corresponding to the user.

Regardless of which options are supplied or any other factor, `renice` shall not 7
alter the system scheduling priorities of any process unless the user requesting 7
such a change has appropriate privileges to do so for the specified process. If the 7
user lacks appropriate privileges to perform the requested action, the utility shall 7
return an error status. 7

On systems where {_POSIX_SAVED_IDS} is defined, the saved set-user-ID of the 7
user's process shall be checked instead of its effective user ID when `renice` 7
attempts to determine the user ID of the process in order to determine whether 7
the user has appropriate privileges. On systems where {_POSIX_SAVED_IDS} is 7
not defined, `renice` shall check the real or effective user ID when attempting to 7
determine the user ID of the process and the user ID of the requester as part of 7
the determination of whether the user has appropriate privileges to perform the 7
requested action. 7

5.24.3 Options

The `renice` utility shall conform to the utility argument syntax guidelines described in 2.10.2. The obsolescent version shall conform with the following exceptions:

— The first operand, *nice_value* , shall precede the options and can have the appearance of a multidigit option.

— The `−g`, `−p`, and `−u` options can each take multiple option-arguments.

— The *pid* option-argument can be used without its `−p` option.

The following options shall be supported by the implementation:

`−g` Interpret all operands (or just the *gid* arguments in the obsolescent version) as unsigned decimal integer process group IDs.

`−n` *increment*
Specify how the system scheduling priority of the specified process(es) shall be adjusted. The *increment* option-argument is a positive or negative decimal integer that shall be used to modify the system scheduling priority of the specified process(es).

Positive *increment* values shall cause a lower system scheduling priority. Negative *increment* values may require appropriate privileges and shall cause a higher system scheduling priority.

`−p` Interpret all operands (or just the *pid* arguments in the obsolescent version) as unsigned decimal integer process IDs. The `−p` option is the default if no options are specified.

`−u` Interpret all operands (or just the *user* arguments in the obsolescent version) as users. If a user exists with a user name equal to the operand, then the user ID of that user shall be used in further processing. Otherwise, if the operand represents an unsigned decimal integer, it shall be used as the numeric user ID of the user.

5.24.4 Operands

The following operands shall be supported by the implementation:

ID A process ID, process group ID, or user name/user ID, depending on the option selected.

nice_value (Obsolescent.) The value specified shall be taken as the actual system scheduling priority, rather than as an increment to the existing system scheduling priority. Specifying a scheduling priority higher than that of the existing process may require appropriate privileges.

5.24.5 External Influences

5.24.5.1 Standard Input

None.

5.24.5.2 Input Files

None.

5.24.5.3 Environment Variables

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

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.

5.24.5.4 Asynchronous Events

Default.

5.24.6 External Effects

5.24.6.1 Standard Output

None.

5.24.6.2 Standard Error

Used only for diagnostic messages.

5.24.6.3 Output Files

None.

5.24.7 Extended Description

None.

5.24.8 Exit Status

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

0	Successful completion.
>0	An error occurred.

5.24.9 Consequences of Errors

Default.

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

Usage, Examples

The *gid*, *pid*, and *user* specifications do not fit either the definition of operand or option-argument. However, for clarity, they have been included in the Options subclause, rather than the Operand subclause.

The following example would adjust the system scheduling priority in such a way that process IDs 987 and 32 would have a lower scheduling priority:

```
renice -n 5 -p 987 32
```

The next example would adjust the system scheduling priority in such a way that group IDs 324 and 76 would have a higher scheduling priority, if the user has the appropriate privileges to do so:

```
renice -n -4 -g 324 76
```

The next example would adjust the system scheduling priority in such a way that numeric user ID 8 and user `sas` would have a lower scheduling priority:

```
renice -n 4 -u 8 sas
```

Useful nice values on historical systems include 19 or 20 (the affected processes will run only when nothing else in the system attempts to run), 0 (the ‘‘base’’ scheduling priority), and any negative number (to make processes run faster).

History of Decisions Made

Originally this utility was written in the historical manner, using the term ‘‘nice value.’’ This was always a point of concern with users, because it was never intuitively obvious what this meant. With the new version of `renice`, which uses the term ‘‘system scheduling priority,’’ it is hoped that novice users will better understand what this utility was meant to do. It also made it much easier to document what the utility was meant to do.

The obsolescent version of `renice` allows pids, gids, and users to be mixed on the command line. This was seen as overly confusing, so when `renice` was rewritten, it was decided to allow only one kind of ID for each invocation. If no `−p`, `−g` or `−u` is specified, any ID operands are considered process IDs.

Strong input was received that the traditional syntax was very difficult to understand and it should be adjusted to comply with the utility syntax guidelines and be more reminiscent of the `nice` utility syntax.

Although this utility has use by system administrators (and in fact appears in the system administration portion of the BSD documentation), the developers of the standard felt that it was very useful for individual end users to control their own processes.

5.25 `split` — Split files into pieces

5.25.1 Synopsis

`split` [−l *line_count*] [−a *suffix_length*] [*file* [*name*]]

`split` −b *n*[k|m] [−a *suffix_length*] [*file* [*name*]]

Obsolescent Version :

`split` [−*line_count*] [−a *suffix_length*] [*file* [*name*]]

5.25.2 Description

The `split` utility shall read an input file and write one or more output files. The default size of each output file shall be 1000 lines. The size of the output files can be modified by specification of the −b or −l options. Each output file shall be created with a unique suffix. The suffix shall consist of exactly *suffix_length* lowercase letters from the POSIX Locale. The letters of the suffix shall be used as if they were a base-26 digit system, with the first suffix to be created consisting of all ‘a’’ characters, the second with a ‘b’’ replacing the last ‘a’’, etc., until a name of all ‘z’’s is created. By default, the names of the output files shall be ‘x’’, followed by a two-character suffix from the character set as described above, starting with aa, ab, ac, etc., and continuing until the suffix zz, for a maximum of 676 files.

If the number of files required exceeds the maximum allowed by the suffix length provided, such that the last allowable file would be larger than the requested size, the `split` utility shall fail after creating the last file with a valid suffix; `split` 7
shall not delete the files it created with valid suffixes. If the file limit is not 7
exceeded, the last file created shall contain the remainder of the input file, and may be smaller than the requested size.

5.25.3 Options

The `split` utility shall conform to the utility argument syntax guidelines described in 2.10.2, except that the obsolescent version allows a multidigit option, −*line_count* .

The following options shall be supported by the implementation:

−a *suffix_length*
: Use *suffix_length* letters to form the suffix portion of the filenames of the split file. If −a is not specified, the default suffix length shall be two. If the sum of the *name* operand and the *suffix_length* option-argument would create a filename exceeding {NAME_MAX} bytes, an error shall result; `split` shall exit with a diagnostic message and no files shall be created.