IEEE P1003.2 Draft 11.2 – September 1991

The `−x` option, historically available only with `fgrep`, is available here for all of the nonobsolescent versions.

4.29 head — Copy the first part of files

4.29.1 Synopsis

`head [−n` *number* `]` [*file* . . .]

Obsolescent version:

`head [−`*number* `]` [*file* . . .]

4.29.2 Description

The `head` utility shall copy its input files to the standard output, ending the output for each file at a designated point.

Copying shall end at the point in each input file indicated by the `−n` *number* option (or the obsolescent version's −*number* argument). The option-argument *number* shall be counted in units of lines.

4.29.3 Options

The `head` utility shall conform to the utility argument syntax guidelines described in standard described in 2.10.2, except that the obsolescent version accepts multicharacter numeric options.

The following option shall be supported by the implementation in the nonobsolescent version:

`−n` *number* The first *number* lines of each input file shall be copied to standard output. The *number* option argument shall be a positive decimal integer.

If no options are specified, `head` shall act as if `-n 10` had been specified.

In the obsolescent version, the following option shall be supported by the implementation:

−*number* The *number* argument is a positive decimal integer with the same effect as the `−n`*number* option in the nonobsolescent version.

4.29.4 Operands

The following operand shall be supported by the implementation:

file A pathname of an input file. If no *file* operands are specified, the standard input shall be used.

4.29.5 External Influences

4.29.5.1 Standard Input

The standard input shall be used only if no *file* operands are specified. See Input Files.

4.29.5.2 Input Files

Input files shall be text files, but the line length shall not be restricted to {LINE_MAX} bytes.

4.29.5.3 Environment Variables

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

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 and input files).

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

4.29.5.4 Asynchronous Events

Default.

4.29.6 External Effects

4.29.6.1 Standard Output

The standard output shall contain designated portions of the input file(s).

If multiple *file* operands are specified, `head` shall precede the output for each with the header:

`"\n==> %s <==\n"`, *<pathname>*

except that the first header written shall not include the initial `<newline>`.

4.29.6.2 Standard Error

Used only for diagnostic messages.

4.29.6.3 Output Files

None.

4.29.7 Extended Description

None.

4.29.8 Exit Status

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

0 Successful completion.

>0 An error occurred.

4.29.9 Consequences of Errors

Default.

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

Usage, Examples

The nonobsolescent version of `head` was created to allow conformance to the Utility Syntax Guidelines. The `-n` option was added to this new interface so that `head` and `tail` would be more logically related.

To write the first ten lines of all files (except those with a leading period) in the directory:

```
head *
```

History of Decisions Made

The `head` utility was not in early drafts. It was felt that `head`, and its frequent companion, `tail`, were useful mostly to interactive users, and not application programs. However, balloting input suggested that these utilities actually do find significant use in scripts, such as to write out portions of log files. Although it is possible to simulate `head` with `sed 10q` for a single file, the working group decided that the popularity of `head` on historical BSD systems warranted its inclusion alongside `tail`.

An earlier draft had the synopsis line:

`head [ −c | −l ] [−n` *number* `] [`*file* ... `]`

This was changed to the current form based on comments and objections noting that `−c` has not been provided by historical versions of `head` and other utilities in POSIX.2 provide similar functionality. Also, `−l` was changed to `−n` to match a similar change in `tail`.

4.30 `id` — Return user identity

4.30.1 Synopsis

`id [`*user* `]`

`id −G [−n] [`*user* `]`

`id −g [−nr] [`*user* `]`

`id −u [−nr] [`*user* `]`

4.30.2 Description

If no *user* operand is provided, the `id` utility shall write the user and group IDs and the corresponding user and group names of the invoking process to standard output. If the effective and real IDs do not match, both shall be written. If multiple groups are supported by the underlying system (see the description of {NGROUPS_MAX} in POSIX.1 {8}), the supplementary group affiliations of the invoking process also shall be written.

If a *user* operand is provided and the process has the appropriate privileges, the user and group IDs of the selected user shall be written. In this case, effective IDs shall be assumed to be identical to real IDs. If the selected user has more than one allowable group membership listed in the group database (see POSIX.1 {8} section 9.1), these shall be written in the same manner as the supplementary groups described in the preceding paragraph.