-- --

IEEE P1003.2 Draft 11.2 − September 1991

4.51 `pwd` — Return working directory name

4.51.1 Synopsis

```
pwd
```

4.51.2 Description

The `pwd` utility shall write an absolute pathname of the current working directory to standard output.

4.51.3 Options

None.

4.51.4 Operands

None.

4.51.5 External Influences

4.51.5.1 Standard Input

None.

4.51.5.2 Input Files

None.

4.51.5.3 Environment Variables

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

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

4.51.5.4 Asynchronous Events

Default.

4.51.6 External Effects

4.51.6.1 Standard Output

The `pwd` utility output shall be an absolute pathname of the current working directory:

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

4.51.6.2 Standard Error

Used only for diagnostic messages.

4.51.6.3 Output Files

None.

4.51.7 Extended Description

None.

4.51.8 Exit Status

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

0 Successful completion.

>0 An error occurred.

4.51.9 Consequences of Errors

If an error is detected, output shall not be written to standard output, a diagnostic message shall be written to standard error, and the exit status shall not be zero.

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

Examples, Usage

Some implementations have historically provided `pwd` as a shell special built-in command.

History of Decisions Made

In most utilities, if an error occurs, partial output may be written to standard output. This does not happen in historical implementations of `pwd`. Because `pwd` is frequently used in existing shell scripts without checking the exit status, it is important that the historical behavior is required here; therefore, the Consequences of Errors subclause specifically disallows any partial output being written to standard output.

4.52 `read` — Read a line from standard input

4.52.1 Synopsis

`read [−r]` *var* . . .

4.52.2 Description

The `read` utility shall read a single line from standard input.

By default, unless the `−r` option is specified, backslash (\) shall act as an escape character, as described in 3.2.1.

The line shall be split into fields (see the definition in 3.1.3) as in the shell (see 3.6.5); the first field shall be assigned to the first variable *var,* the second field to the second variable *var,* etc. If there are fewer *var* operands specified than there are fields, the leftover fields and their intervening separators shall be assigned to the last *var*. If there are fewer fields than *var* s, the remaining *var* s shall be set to empty strings.

The setting of variables specified by the *var* operands shall affect the current shell execution environment; see 3.12.

4.52.3 Options

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

The following option shall be supported by the implementation: