cd(1p) — Linux manual page

PROLOG top

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME top

cd — change the working directory

SYNOPSIS top

```
cd [-L|-P] [directory]

cd -
```

DESCRIPTION top

The *cd* utility shall change the working directory of the current shell execution environment (see *Section 2.12*, *Shell Execution Environment*) by executing the following steps in sequence. (In the following steps, the symbol **curpath** represents an intermediate value used to simplify the description of the algorithm used by *cd*. There is no requirement that **curpath** be made visible to the application.)

1. If no *directory* operand is given and the *HOME* environment variable is empty or undefined, the default behavior is implementation-defined and no further steps shall be taken.

2. If no *directory* operand is given and the *HOME* environment variable is set to a non-empty value, the *cd* utility shall behave as if the directory named in the *HOME* environment variable was specified as the *directory* operand.

3. If the *directory* operand begins with a <slash> character, set **curpath** to the operand and proceed to step 7.

4. If the first component of the *directory* operand is dot or dot-dot, proceed to step 6.

5. Starting with the first pathname in the <colon>-separated pathnames of *CDPATH* (see the ENVIRONMENT VARIABLES section) if the pathname is non-null, test if the concatenation of that pathname, a <slash> character if that pathname did not end with a <slash> character, and the *directory* operand names a directory. If the pathname is null, test if the concatenation of dot, a <slash> character, and the operand names a directory. In either case, if the resulting string names an existing directory, set **curpath** to that string and proceed to step 7. Otherwise, repeat this step with the next pathname in *CDPATH* until all pathnames have been tested.

6. Set **curpath** to the *directory* operand.

7. If the **-P** option is in effect, proceed to step 10. If **curpath** does not begin with a <slash> character, set **curpath** to the string formed by the concatenation of the value of *PWD*, a <slash> character if the value of *PWD* did not end with a <slash> character, and **curpath**.

8. The **curpath** value shall then be converted to canonical form as follows, considering each component from beginning to end, in sequence:

    a. Dot components and any <slash> characters that separate them from the next component shall be deleted.

    b. For each dot-dot component, if there is a preceding component and it is neither root nor dot-dot, then:

        i. If the preceding component does not refer (in the context of pathname resolution with symbolic links followed) to a directory, then the *cd* utility shall display an appropriate error message and no further steps shall be taken.

        ii. The preceding component, all <slash> characters separating the preceding component from dot-dot, dot-dot, and all <slash> characters separating dot-dot from the following component (if any) shall be deleted.

c. An implementation may further simplify **curpath** by removing any trailing <slash> characters that are not also leading <slash> characters, replacing multiple non-leading consecutive <slash> characters with a single <slash>, and replacing three or more leading <slash> characters with a single <slash>. If, as a result of this canonicalization, the **curpath** variable is null, no further steps shall be taken.

9. If **curpath** is longer than {PATH_MAX} bytes (including the terminating null) and the *directory* operand was not longer than {PATH_MAX} bytes (including the terminating null), then **curpath** shall be converted from an absolute pathname to an equivalent relative pathname if possible. This conversion shall always be considered possible if the value of *PWD*, with a trailing <slash> added if it does not already have one, is an initial substring of **curpath**. Whether or not it is considered possible under other circumstances is unspecified. Implementations may also apply this conversion if **curpath** is not longer than {PATH_MAX} bytes or the *directory* operand was longer than {PATH_MAX} bytes.

10. The *cd* utility shall then perform actions equivalent to the *chdir*() function called with **curpath** as the *path* argument. If these actions fail for any reason, the *cd* utility shall display an appropriate error message and the remainder of this step shall not be executed. If the **-P** option is not in effect, the *PWD* environment variable shall be set to the value that **curpath** had on entry to step 9 (i.e., before conversion to a relative pathname). If the **-P** option is in effect, the *PWD* environment variable shall be set to the string that would be output by *pwd* **-P**. If there is insufficient permission on the new directory, or on any parent of that directory, to determine the current working directory, the value of the *PWD* environment variable is unspecified.

If, during the execution of the above steps, the *PWD* environment variable is set, the *OLDPWD* environment variable shall also be set to the value of the old working directory (that is the current working directory immediately prior to the call to *cd*).

OPTIONS top

The *cd* utility shall conform to the Base Definitions volume of POSIX.1-2017, *Section 12.2*, *Utility Syntax Guidelines*.

The following options shall be supported by the implementation:

-L Handle the operand dot-dot logically; symbolic link

components shall not be resolved before dot-dot components are processed (see steps 8. and 9. in the DESCRIPTION).

-P Handle the operand dot-dot physically; symbolic link components shall be resolved before dot-dot components are processed (see step 7. in the DESCRIPTION).

If both **-L** and **-P** options are specified, the last of these options shall be used and all others ignored. If neither **-L** nor **-P** is specified, the operand shall be handled dot-dot logically; see the DESCRIPTION.

OPERANDS top

The following operands shall be supported:

directory An absolute or relative pathname of the directory that shall become the new working directory. The interpretation of a relative pathname by *cd* depends on the **-L** option and the *CDPATH* and *PWD* environment variables. If *directory* is an empty string, the results are unspecified.

- When a <hyphen-minus> is used as the operand, this shall be equivalent to the command:

```
cd "$OLDPWD" && pwd
```

which changes to the previous working directory and then writes its name.

STDIN top

Not used.

INPUT FILES top

None.

ENVIRONMENT VARIABLES top

The following environment variables shall affect the execution of *cd*:

CDPATH A <colon>-separated list of pathnames that refer to directories. The *cd* utility shall use this list in its attempt to change the directory, as described in the

DESCRIPTION. An empty string in place of a directory pathname represents the current directory. If *CDPATH* is not set, it shall be treated as if it were an empty string.

HOME The name of the directory, used when no *directory* operand is specified.

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of POSIX.1-2017, *Section 8.2*, *Internationalization Variables* for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES
Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard error.

NLSPATH Determine the location of message catalogs for the processing of *LC_MESSAGES*.

OLDPWD A pathname of the previous working directory, used by *cd* **-**.

PWD This variable shall be set as specified in the DESCRIPTION. If an application sets or unsets the value of *PWD*, the behavior of *cd* is unspecified.

ASYNCHRONOUS EVENTS top

Default.

STDOUT top

If a non-empty directory name from *CDPATH* is used, or if *cd* **-** is used, an absolute pathname of the new working directory shall be written to the standard output as follows:

"%s\n", <*new directory*>

Otherwise, there shall be no output.

STDERR top

The standard error shall be used only for diagnostic messages.

OUTPUT FILES top

None.

EXTENDED DESCRIPTION top

None.

EXIT STATUS top

The following exit values shall be returned:

0 The directory was successfully changed.

>0 An error occurred.

CONSEQUENCES OF ERRORS top

The working directory shall remain unchanged.

The following sections are informative.

APPLICATION USAGE top

Since *cd* affects the current shell execution environment, it is always provided as a shell regular built-in. If it is called in a subshell or separate utility execution environment, such as one of the following:

```
(cd /tmp)
nohup cd
find . -exec cd {} \;
```

it does not affect the working directory of the caller's environment.

The user must have execute (search) permission in *directory* in order to change to it.

EXAMPLES top

The following template can be used to perform processing in the directory specified by *location* and end up in the current working directory in use before the first *cd* command was issued:

```
cd location
if [ $? -ne 0 ]
then
    print error message
    exit 1
fi
... do whatever is desired as long as the OLDPWD environment variable
    is not modified
cd -
```

RATIONALE top

The use of the *CDPATH* was introduced in the System V shell. Its use is analogous to the use of the *PATH* variable in the shell. The BSD C shell used a shell parameter *cdpath* for this purpose.

A common extension when *HOME* is undefined is to get the login directory from the user database for the invoking user. This does not occur on System V implementations.

Some historical shells, such as the KornShell, took special actions when the directory name contained a dot-dot component, selecting the logical parent of the directory, rather than the actual parent directory; that is, it moved up one level toward the '/' in the pathname, remembering what the user typed, rather than performing the equivalent of:

```
chdir("..");
```

In such a shell, the following commands would not necessarily produce equivalent output for all directories:

```
cd .. && ls      ls ..
```

This behavior is now the default. It is not consistent with the definition of dot-dot in most historical practice; that is, while this behavior has been optionally available in the KornShell, other shells have historically not supported this functionality. The logical pathname is stored in the *PWD* environment variable when the *cd* utility completes and this value is used to construct the next directory name if *cd* is invoked with the **-L** option.

FUTURE DIRECTIONS top

None.

SEE ALSO top

Section 2.12, *Shell Execution Environment*, pwd(1p)

The Base Definitions volume of POSIX.1-2017, *Chapter 8*, *Environment Variables*, *Section 12.2*, *Utility Syntax Guidelines*

The System Interfaces volume of POSIX.1-2017, chdir(3p)

COPYRIGHT top

Pages that refer to this page: pwd(1p), sh(1p)