

Full credit is given to the above companies including the OS that this PDF file was generated!

# Red Hat Enterprise Linux Release 9.2 Manual Pages on 'wordexp.3' command

# \$ man wordexp.3

WORDEXP(3)

Linux Programmer's Manual

WORDEXP(3)

NAME

wordexp, wordfree - perform word expansion like a posix-shell

# SYNOPSIS

#include <wordexp.h>

int wordexp(const char \*s, wordexp\_t \*p, int flags);

void wordfree(wordexp\_t \*p);

Feature Test Macro Requirements for glibc (see feature test macros(7)):

wordexp(), wordfree(): \_XOPEN\_SOURCE

#### DESCRIPTION

The function wordexp() performs a shell-like expansion of the string s and returns the result in the structure pointed to by p. The data type wordexp\_t is a structure that at least has the fields we\_wordc, we\_wordv, and we\_offs. The field we\_wordc is a size\_t that gives the number of words in the expansion of s. The field we\_wordv is a char \*\* that points to the array of words found. The field we\_offs of type size\_t is sometimes (depending on flags, see below) used to indicate the number of initial elements in the we\_wordv array that should be filled with NULLs.

The function wordfree() frees the allocated memory again. More pre? cisely, it does not free its argument, but it frees the array we\_wordv and the strings that points to.

The string argument Page 1/5

Since the expansion is the same as the expansion by the shell (see sh(1)) of the parameters to a command, the string s must not contain characters that would be illegal in shell command parameters. In par? ticular, there must not be any unescaped newline or |, &, ;, <, >, (, ),  $\{$ ,  $\}$  characters outside a command substitution or parameter substi? tution context.

If the argument s contains a word that starts with an unquoted comment character #, then it is unspecified whether that word and all following words are ignored, or the # is treated as a non-comment character.

### The expansion

The expansion done consists of the following stages: tilde expansion (replacing ~user by user's home directory), variable substitution (re? placing \$FOO by the value of the environment variable FOO), command substitution (replacing \$(command) or `command` by the output of com? mand), arithmetic expansion, field splitting, wildcard expansion, quote removal.

The result of expansion of special parameters (\$@, \$\*, \$#, \$?, \$-, \$\$, \$!, \$0) is unspecified.

Field splitting is done using the environment variable \$IFS. If it is not set, the field separators are space, tab and newline.

#### The output array

The array we\_wordv contains the words found, followed by a NULL.

### The flags argument

The flag argument is a bitwise inclusive OR of the following values:

## WRDE\_APPEND

Append the words found to the array resulting from a previous call.

## WRDE\_DOOFFS

Insert we\_offs initial NULLs in the array we\_wordv. (These are not counted in the returned we wordc.)

## WRDE\_NOCMD

Don't do command substitution.

WRDE\_REUSE Page 2/5

The argument p resulted from a previous call to wordexp(), and wordfree() was not called. Reuse the allocated storage.

#### WRDE SHOWERR

Normally during command substitution stderr is redirected to /dev/null. This flag specifies that stderr is not to be redi? rected.

### WRDE\_UNDEF

Consider it an error if an undefined shell variable is expanded.

### **RETURN VALUE**

In case of success 0 is returned. In case of error one of the follow? ing five values is returned.

### WRDE\_BADCHAR

Illegal occurrence of newline or one of |, &, ;, <, >, (, ),  $\{$ ,  $\}$ .

### WRDE\_BADVAL

An undefined shell variable was referenced, and the WRDE\_UNDEF flag told us to consider this an error.

#### WRDE CMDSUB

Command substitution requested, but the WRDE\_NOCMD flag told us to consider this an error.

#### WRDE\_NOSPACE

Out of memory.

### WRDE\_SYNTAX

Shell syntax error, such as unbalanced parentheses or unmatched quotes.

## **VERSIONS**

wordexp() and wordfree() are provided in glibc since version 2.1.

### **ATTRIBUTES**

For an explanation of the terms used in this section, see at? tributes(7).

?Interface ? Attribute ? Value

?

```
?wordexp() ? Thread safety ? MT-Unsafe race:utent const:env ?
                   ? env sig:ALRM timer locale
   ?wordfree() ? Thread safety ? MT-Safe
   In the above table, utent in race:utent signifies that if any of the
   functions setutent(3), getutent(3), or endutent(3) are used in parallel
   in different threads of a program, then data races could occur. word?
   exp() calls those functions, so we use race:utent to remind users.
CONFORMING TO
   POSIX.1-2001, POSIX.1-2008.
EXAMPLES
   The output of the following example program is approximately that of
   "ls [a-c]*.c".
   #include <stdio.h>
   #include <stdlib.h>
   #include <wordexp.h>
   int
   main(int argc, char **argv)
   {
     wordexp_t p;
     char **w;
     wordexp("[a-c]*.c", &p, 0);
     w = p.we_wordv;
     for (int i = 0; i < p.we_wordc; i++)
       printf("%s\n", w[i]);
     wordfree(&p);
     exit(EXIT_SUCCESS);
   }
SEE ALSO
   fnmatch(3), glob(3)
COLOPHON
```

description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man-pages/.

2020-11-01 WORDEXP(3)