NAME

getopt — get option character from command line argument list

LIBRARY

Standard C Library (libc, −lc)

SYNOPSIS

```
#include <unistd.h>

extern char *optarg;
extern int optind;
extern int optopt;
extern int opterr;
extern int optreset;

int
getopt(int argc, char * const *argv, const char *optstring);
```

DESCRIPTION

The **getopt**() function incrementally parses a command line argument list *argv* and returns the next *known* option character. An option character is *known* if it has been specified in the string of accepted option characters, *optstring*.

The option string *optstring* may contain the following elements: individual characters, and characters followed by a colon to indicate an option argument is to follow. For example, an option string "x" recognizes an option "**−x**", and an option string "x:" recognizes an option and argument "**−x** *argument*". It does not matter to **getopt**() if a following argument has leading white space.

On return from **getopt**(), *optarg* points to an option argument, if it is anticipated, and the variable *optind* contains the index to the next *argv* argument for a subsequent call to **getopt**(). The variable *optopt* saves the last *known* option character returned by **getopt**().

The variable *opterr* and *optind* are both initialized to 1. The *optind* variable may be set to another value before a set of calls to **getopt**() in order to skip over more or less argv entries.

In order to use **getopt**() to evaluate multiple sets of arguments, or to evaluate a single set of arguments multiple times, the variable *optreset* must be set to 1 before the second and each additional set of calls to **getopt**(), and the variable *optind* must be reinitialized.

The **getopt**() function returns −1 when the argument list is exhausted, or '?' if a non-recognized option is encountered. The interpretation of options in the argument list may be cancelled by the option '--' (double dash) which causes **getopt**() to signal the end of argument processing and return −1. When all options have been processed (i.e., up to the first non-option argument), **getopt**() returns −1.

DIAGNOSTICS

If the **getopt**() function encounters a character not found in the string *optstring* or detects a missing option argument it writes an error message to the stderr and returns '?'. Setting *opterr* to a zero will disable these error messages. If *optstring* has a leading ':' then a missing option argument causes a ':' to be returned in addition to suppressing any error messages.

Option arguments are allowed to begin with "−"; this is reasonable but reduces the amount of error checking possible.

EXTENSIONS

The *optreset* variable was added to make it possible to call the **getopt**() function multiple times. This is an extension to the IEEE Std 1003.2 ("POSIX.2") specification.

EXAMPLES

```
int bflag, ch, fd;

bflag = 0;
while ((ch = getopt(argc, argv, "bf:")) != -1)
       switch (ch) {
       case 'b':
              bflag = 1;
              break;
       case 'f':
              if ((fd = open(optarg, O_RDONLY, 0)) < 0)
                     err(1, "%s", optarg);
              break;
       case '?':
       default:
              usage();
       }
argc -= optind;
argv += optind;
```

HISTORY

The **getopt**() function appeared in 4.3BSD.

BUGS

The **getopt**() function was once specified to return EOF instead of −1. This was changed by IEEE Std 1003.2-1992 ("POSIX.2") to decouple **getopt**() from <stdio.h>.

A single dash "-" may be specified as a character in *optstring*, however it should *never* have an argument associated with it. This allows **getopt**() to be used with programs that expect "-" as an option flag. This practice is wrong, and should not be used in any current development. It is provided for backward compatibility *only*. By default, a single dash causes **getopt**() to return −1. This is, we believe, compatible with System V.

It is also possible to handle digits as option letters. This allows **getopt**() to be used with programs that expect a number ("-3") as an option. This practice is wrong, and should not be used in any current development. It is provided for backward compatibility *only*. The following code fragment works in most (but not all) cases.

```
int length;
char *p, *ep;

while ((ch = getopt(argc, argv, "0123456789")) != -1)
       switch (ch) {
       case '0': case '1': case '2': case '3': case '4':
       case '5': case '6': case '7': case '8': case '9':
              p = argv[optind - 1];
              if (p[0] == '-' && p[1] == ch && !p[2])
                     length = strtol(++p, &ep, 10);
              else if (argv[optind] && argv[optind][1] == ch) {
```

```
                        length = strtol((p = argv[optind] + 1),
                            &ep, 10);
                        optind++;
                        optreset = 1;
                } else
                        usage();
                if (*ep != ' ')
                        errx(EX_USAGE, "illegal number -- %s", p);
                break;
        }
```