

getopt - Man Page

command option parsing

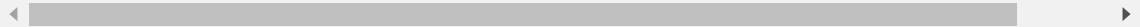
Prolog

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.

Synopsis

```
#include <unistd.h>
```

```
int getopt(int argc, char * const argv[], const char *optstr);  
extern char *optarg;  
extern int opterr, optind, optopt;
```



Description

The *getopt()* function is a command-line parser that shall follow Utility Syntax Guidelines 3, 4, 5, 6, 7, 9, and 10 in the Base Definitions volume of POSIX.1-2017, *Section 12.2, Utility Syntax Guidelines*.

The parameters *argc* and *argv* are the argument count and argument array as passed to *main()* (see *exec()*). The argument *optstring* is a string of recognized option characters; if a character is followed by a <colon>, the option takes an argument. All option characters allowed by Utility Syntax Guideline 3 are allowed in *optstring*. The implementation may accept other characters as an extension.

getopt - Man Page

getopt() shall update *ic* when it finishes with each element of *argv*.

If the application sets *optind* to zero before calling *getopt()*, the behavior is unspecified. When an element of *argv* contains multiple option characters, it is unspecified how *getopt()* determines which options have already been processed.

The *getopt()* function shall return the next option character (if one is found) from *argv* that matches a character in *optstring*, if there is one that matches. If the option takes an argument, *getopt()* shall set the variable *optarg* to point to the option-argument as follows:

1. If the option was the last character in the string pointed to by an element of *argv*, then *optarg* shall contain the next element of *argv*, and *optind* shall be incremented by 2. If the resulting value of *optind* is greater than *argc*, this indicates a missing option-argument, and *getopt()* shall return an error indication.
2. Otherwise, *optarg* shall point to the string following the option character in that element of *argv*, and *optind* shall be incremented by 1.

If, when *getopt()* is called:

```
argv[optind]  is a null pointer
*argv[optind] is not the character '-'
argv[optind]  points to the string "--"
```

getopt() shall return -1 without changing *optind*. If:

```
argv[optind]  points to the string "--"
```

getopt() shall return -1 after incrementing *optind*.

If *getopt()* encounters an option character that is not contained in *optstring*, it shall return the <question-mark> ('?') character. If it detects a missing option-argument, it shall return the <colon>

getopt - Man Page

shall set the variable *optopt* to the option character that caused the error. If the application has not set the variable *opterr* to 0 and the first character of *optstring* is not a <colon>, *getopt()* shall also print a diagnostic message to *stderr* in the format specified for the *getopts* utility, unless the stream has wide orientation, in which case the behavior is undefined.

The *getopt()* function need not be thread-safe.

Return Value

The *getopt()* function shall return the next option character specified on the command line.

<colon> (':') возвращается, если *getopt()* обнаруживает отсутствующий аргумент и первым символом *optstring* был <colon> (':').

<Вопросительный знак> ('?') возвращается, если *getopt()* встречает символ опции, отсутствующий в *optstring*, или обнаруживает отсутствующий аргумент, а первый символ *optstring* не был <двоеточием> (':').

В противном случае *getopt()* возвращает -1 при анализе всех параметров командной строки.

Ошибки

Если приложение не установило переменную *opterr* в 0, первый символ *optstring* не является <двоеточием>, и возникает ошибка записи, когда *getopt()* печатает диагностическое сообщение в *stderr*, то индикатор ошибки для *stderr* должен быть установлен; но *getopt()* все равно будет успешным, и ошибка записи будет удалена. значение *errno* после *getopt()* не определено.

getopt - Man Page

Примеры

Синтаксический анализ параметров командной строки

Следующий фрагмент кода показывает, как можно обрабатывать аргументы для утилиты, которая может принимать взаимоисключающие параметры *a* и *b* и параметры *f* и *o*, оба из которых требуют аргументов:

```
#включить <stdio.h>
#включить <stdlib.h>
#include <unistd.h>

int
main(int argc, char *argv[ ])
{
    int c;
    int bflg = 0, aflag = 0, errflag = 0;
    char *ifile;
    char *ofile;

    while ((c = getopt(argc, argv, ":abf:o:")) != -1) {
        switch(c) {
        case 'a':
            if (bflg)
                errflag++;
            else
                aflag++;
            перерыв;
        case 'b':
            if (aflag)
                errflag++;
            else
                bflag++;
            перерыв;
        case 'f':
```

getopt - Man Page

```
ofile = optarg;
перерыв;
case '::': /* -f или -o без операнда */
    fprintf(stderr,
"Option -%c требует операнда\n", optopt);
    errflg++;
    перерыв;
case '?':
    fprintf(stderr,
"Unrecognized option: '-%c'\n", optopt);
    errflg++;
    }
}
if (errflg) {
    fprintf(stderr, "использование: ... ");
    ВЫХОД (2);
}
for ( ; optind < argc; optind++) {
    if (access(argv[optind], R_OK)) {
        . . .
    }
}
```

Этот код принимает любое из следующих значений в качестве эквивалента:

```
cmd -ao arg path path
cmd -a -o arg path path
cmd -o arg -a path path
cmd -a -o arg -- path path
cmd -a -oarg path path
cmd -aoarg path path
```

Выбор параметров из командной строки

В следующем примере выбирается тип подпрограмм базы данных, которые пользователь хочет использовать, на основе аргумента *Options* .

getopt - Man Page

```
const char *Options = "hdbtl";
...
int dbtype, c;
char *st;
...
dbtype = 0;
while ((c = getopt(argc, argv, Options)) != -1) {
if ((st = strchr(Options, c)) != NULL) {
    dbtype = st - Options;
    break;
}
}
```

Использование приложений

Функция *getopt()* требуется только для поддержки символов опций, включенных в Руководство по синтаксису утилиты 3. Многие исторические реализации *getopt()* поддерживают другие символы в качестве опций. Это разрешенное расширение, но приложения, использующие расширения, не являются максимально переносимыми. Обратите внимание, что поддержка многобайтовых символов опций возможна только в том случае, если такие символы могут быть представлены как тип **int**.

Приложения, которые используют широкосимвольные функции вывода с *stderr*, должны гарантировать, что любые вызовы *getopt()* не записываются в *stderr*, либо установив *opterr* в 0, либо гарантируя, что первый символ *optstring* всегда является <двоеточием> .

Хотя *ferror(stderr)* может использоваться для обнаружения сбоев при записи диагностики в *stderr*, когда *getopt()* возвращает '?', значение *errno* в таком состоянии не указано. Приложения, желающие получить больше контроля над обработкой сбоев записи, должны установить *opterr* в 0 и независимо выполнять вывод в *stderr*, а не полагаться на *getopt()* для вывода.

getopt - Man Page

Переменная `optopt` представляет историческую практику и позволяет приложению получить идентификатор недопустимой опции.

Описание было написано, чтобы дать понять, что *getopt()*, как и утилита *getopts*, имеет дело с аргументами `option`, независимо от того, отделены ли они от опции символами `<blank>` или нет. Обратите внимание, что требования к *getopt()* и *getopts* более строгие, чем рекомендации по синтаксису утилиты.

Функция *getopt()* должна возвращать `-1`, а не `EOF`, так что `<stdio.h>` не требуется.

Особое значение `<двоеточия>` в качестве первого символа *optstring* делает *getopt()* совместимым с утилитой *getopts*. Это позволяет приложению проводить различие между отсутствующим аргументом и неправильной буквой опции без необходимости проверять букву опции. Это правда, что отсутствующий аргумент может быть обнаружен только в одном случае, но это случай, который необходимо учитывать.

Будущие направления

Нет.

См. Также

[exec](#)

Базовый том определений POSIX.1-2017, *Раздел 12.2, Руководство по синтаксису утилит*, [<unistd.h>](#)

Объем оболочки и утилит POSIX.1-2017, [getopts](#)

getopt - Man Page

Части этого текста перепечатаны и воспроизведены в электронном виде из IEEE Std 1003.1-2017, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 7, 2018 Edition, Copyright (C) 2018 by the Institute of Electrical and Electronics Engineers, Inc and The Open GroupГруппа. В случае любого несоответствия между этой версией и исходным стандартом IEEE и Open Group исходный стандарт IEEE и Open Group является документом рефери. Оригинальный стандарт можно получить онлайн по адресу <http://www.opengroup.org/unix/online.html> .

Любые типографские ошибки или ошибки форматирования, которые появляются на этой странице, скорее всего, были введены во время преобразования исходных файлов в формат man page. Чтобы сообщить о таких ошибках, см. https://www.kernel.org/doc/man-pages/reporting_bugs.html .

Ссылка на

`getopts (1p)`, `getsubopt (3p)`, `stdio.h(0p)`, `unistd.h(0p)`.

2017 IEEE/The Open Group POSIX Programmer's Manual

getopt - Man Page

[Главная](#) [Блог](#) [0 нас](#)