

## ***Функции для разбора списка переданных опций:***

*int getopt()* - Обрабатывает короткие опции

*int getopt\_long()* - Обрабатывает короткие и длинные опции

*int getopt\_long\_only()* - Обрабатывает опции только как длинные

# Функция `getopt()`

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

```
int getopt(  
    int argc, /* число параметров программы*/  
    char * const argv[], /* массив указателей на эти параметры*/  
    const char *optstring /* строка с допустимыми символами опций. */  
);
```

```
extern char *optarg; /* указатель аргумент опции, если таковой имеется  
*/
```

```
extern int optind, /* индекс указателя argv, который будет обработан при  
    следующем вызове getopt() */  
    opterr, /* управление выводом сообщений об ошибках*/  
    optopt; /* текущая опция*/
```

*optstring* - строка, содержащая допустимые символы опций.

Если за символом опции в строке стоит двоеточие, то опция требует указания аргумента.

Два двоеточия означают, что опция имеет необязательный аргумент (дополнение *GNU*).

Если *optstring* содержит *W*, за которой следует точка с запятой, то *-W foo* рассматривается как длинная опция *--foo*.

Если первым символом *optstring* является "+" или задана переменная окружения *POSIXLY\_CORRECT*, то обработка опций прерывается на первом аргументе, не являющемся опцией.

Если первым символом *optstring* является "-", то каждый элемент *argv*, не являющийся опцией, обрабатывается так, как если бы он был аргументом опции с символом, имеющим код 1.

При каждом повторном вызове *getopt()* возвращаются символы следующей опции. Если *getopt()* находит символ опции, она возвращает этот символ, обновляя внешнюю переменную *optind*, так что следующий вызов *getopt()* может продолжить проверку со следующего символа опции или элемента *argv*.

Если символов опций больше нет, то *getopt()* возвращает -1. При этом *optind* станет индексом первого элемента *argv*, не являющегося опцией (*getopt()* переставляет элементы содержимого *argv* в процессе поиска, так что все аргументы, не являющиеся опциями, оказываются в конце).

Если *getopt()* не распознал символ опции, то он выводит на стандартный поток ошибок соответствующее сообщение, заносит символ в *optopt* и возвращает "?". Вызывающая программа может предотвратить вывод сообщения об ошибке, установив нулевое значение *opterr*.

Если функция *getopt()* обнаружит в *argv* символ опции, не найденный в *optstring*, или если она обнаружит пропущенный аргумент опции, то она возвратит '?' и занесет во внешнюю переменную *optopt* действительный символ опции. Если первым символом *optstring* является двоеточие (':'), то для сообщения о пропущенном аргументе опции *getopt()* возвратит ':' вместо '?'. Если при обнаружении ошибки первый символ *optstring* не является двоеточием и внешняя переменная *opterr* не равно нулю (по умолчанию так и есть), то *getopt()* выведет сообщение об ошибке.

```

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

int main (int argc, char *argv[])
{
    int c;
    char* s[2] = {"Hello", "By"};
    char nm[] = "Incognito";
    struct globalArgs_t {
        int helloby; /* опция -g (-b) */
        char *name; /* опция -n */
        int errflag;
    } globalArgs = {0, nm, 0};
    while ((c=getopt (argc, argv, ":gbn:")) != -1)
        switch (c)
        {
            case 'g' : globalArgs.helloby = 0; break;
            case 'b' : globalArgs.helloby = 1; break;
            case 'n' : globalArgs.name = optarg; break;
            case '?' : globalArgs.errflag =1; printf("Wrong option %c\n", optopt); break;
            case ':' : globalArgs.errflag =1; printf("Option %c is missed\n", optopt); break;
        }
    if (globalArgs.errflag == 1)
        return EXIT_FAILURE;

    printf("%s, %s", s[globalArgs.helloby], globalArgs.name);
    if (optind < argc)
        printf("\nOther arguments:\n");
    for ( ; optind < argc; optind++)
        printf("%s ", argv[optind]);
    printf("\n");
    return EXIT_SUCCESS;
}

```

```
nt@nata-pc:~$ ./getoptexample
Hello, Incognito
nt@nata-pc:~$ ./getoptexample -h
Wrong option h
nt@nata-pc:~$ ./getoptexample -g
Hello, Incognito
nt@nata-pc:~$ ./getoptexample -b
By, Incognito
nt@nata-pc:~$ ./getoptexample -b -h
Wrong option h
nt@nata-pc:~$ ./getoptexample -b -g -n
Option n argument is missed
nt@nata-pc:~$ ./getoptexample -b -g
By, Incognito
nt@nata-pc:~$ ./getoptexample -b -g -nGroup
By, Group
nt@nata-pc:~$ ./getoptexample -b -g -nGroup fff
By, Group
Other arguments:
fff
nt@nata-pc:~$ ./getoptexample ddd ddd sss -b -g
-nGroup
By, Group
Other arguments:
ddd ddd sss
```

# Функции *getopt\_long()* и *getopt\_long\_only()*

```
#define _GNU_SOURCE
```

```
#include <getopt.h>
```

```
int getopt_long(  
    int argc,  
    char * const argv[],  
    const char *optstring,  
    const struct option *longopts,  
    int *longindex  
);
```

```
int getopt_long_only(i  
    int argc,  
    char * const argv[],  
    const char *optstring,  
    const struct option *longopts,  
    int *longindex  
);
```

Функция *getopt\_long()* работает так же, как *getopt()*, за исключением того, что она воспринимает и длинные опции, начинающиеся с двух дефисов. Длинные опции можно сокращать, если сокращение сохраняет уникальность опции или полностью совпадает с одной из определенных опций. Длинная опция может иметь параметр вида --опция=параметр или --опция параметр.

Функция *getopt\_long\_only()* работает так же, как *getopt\_long()*, но в качестве указателя длинной опции может служить не только "--", но и "-". Если опция, начинающаяся с "-" (не с "--"), не совпадает с длинной опцией, но совпадает с короткой, то она обрабатывается как короткая опция.

*longopts* является указателем на первый элемент массива структур `struct option`, указанного в `<getopt.h>` следующим образом

```
struct option {
    const char *name;
    int has_arg;
    int *flag;
    int val;
};
```

Значения различных полей:

*name* - имя длинной опции.

*has\_arg* может быть:

- *no\_argument* (или 0), если опция не имеет аргумента;
- *required\_argument* (или 1), если опция требует указания аргумента;
- *optional\_argument* (или 2), если опция может иметь необязательный аргумент.

*flag* задает способ возвращения результатов для длинной опции. Если *flag* равен `NULL`, то `getopt_long()` возвращает *val*. Например, вызывающая программа может назначить *val* эквивалентом символа короткой опции. В противном случае `getopt_long()` возвращает 0, а *flag* указывает на переменную, устанавливаемое значение которой равно *val*, если опция найдена; и оставляемую без изменений, если опция не найдена.

*val* является возвращаемым значением или загружается в переменную, на которую указывает *flag*.

Последний элемент массива должен быть заполнен нулями.

Если *longindex* не содержит `NULL`, то он является указателем на переменную, содержащую индекс длинной опции в соответствии с *longopts*.

```

#include <stdlib.h>
#include <stdio.h>
#define _GNU_SOURCE
#include <getopt.h>
int main (int argc, char *argv[])
{
    int c;
    char* s[2] = {"Hello","By"};
    char nm[] = "Incognito";
    struct globalArgs_t {
        int helloby; /* опция -g (-b) */
        char *name; /* опция -n */
        int errflag;
    } globalArgs = {0,nm,0};

    struct option long_options[] = {
        { "hello", no_argument, 0, 'g' },
        { "by", no_argument, 0, 'b' },
        { "name", required_argument, 0, 'n' },
        {NULL,0,NULL,0}
    };
    int option_index = 0;
    while ((c=getopt_long(argc,argv,":gbn:",long_options, &option_index )) != -1)
        switch (c)
        {
            case 'g' : globalArgs.helloby = 0; break;
            case 'b' : globalArgs.helloby = 1; break;
            case 'n' : globalArgs.name = optarg; break;
            case '?' : globalArgs.errflag =1; printf("Wrong option %c\n", optopt); break;
            case ':' : globalArgs.errflag =1; printf("Option %c argument is missed\n", optopt); break;
        }
    if (globalArgs.errflag == 1)
        return EXIT_FAILURE;
    printf("%s, %s", s[globalArgs.helloby],globalArgs.name);
    if (optind < argc)
        printf("\nOther arguments:\n");
    for ( ; optind < argc; optind++)
        printf("%s ", argv[optind]);
    printf("\n");

    return EXIT_SUCCESS;
}

```

```
nt@nata-pc:~$ ./golong -b
By, Incognito
nt@nata-pc:~$ ./golong -b -g
Hello, Incognito
nt@nata-pc:~$ ./golong -b -g -n
Option n argument is missed
nt@nata-pc:~$ ./golong -b -hello
Wrong option h
Wrong option e
Wrong option l
Wrong option l
Wrong option o
nt@nata-pc:~$ ./golong -b --hello
Hello, Incognito
nt@nata-pc:~$ ./golong -b --hello --by
By, Incognito
nt@nata-pc:~$ ./golong -b --hello --name=Group
Hello, Group
nt@nata-pc:~$ ./golong -b --hello aaa
--name=Group bbb
Hello, Group
Other arguments:
aaa bbb
```