

[Каталог документации](#) / [Раздел "Программирование, языки"](#) / [Оглавление документа](#)

Advanced Bash-Scripting Guide: Искусство программирования на языке сценариев командной оболочки

[Назад](#)

Глава 7. Проверка условий

[Вперед](#)

7.2. Операции проверки файлов

Возвращает true если...

-e

файл существует

-f

обычный файл (не каталог и не файл устройства)

-s

ненулевой размер файла

-d

файл является каталогом

-b

файл является блочным устройством (floppy, cdrom и т.п.)

-c

файл является символьным устройством (клавиатура, модем, звуковая карта и т.п.)

-p

файл является каналом

-h

файл является символической ссылкой

-L

файл является символической ссылкой

-S

файл является сокетом

-t

файл (**дескриптор**) связан с терминальным устройством

Этот ключ может использоваться для проверки -- является ли файл стандартным устройством ввода `stdin` (`[-t 0]`) или стандартным устройством вывода `stdout` (`[-t 1]`).

-r

файл доступен для чтения (*пользователю, запустившему сценарий*)

-w

файл доступен для записи (*пользователю, запустившему сценарий*)

-x

файл доступен для исполнения (*пользователю, запустившему сценарий*)

-g

`set-group-id` (`sgid`) флаг для файла или каталога установлен

Если для каталога установлен флаг *sgid*, то файлы, создаваемые в таком каталоге, наследуют идентификатор группы каталога, который может не совпадать с идентификатором группы, к которой принадлежит пользователь, создавший файл. Это может быть полезно для каталогов, в которых хранятся файлы, общедоступные для группы пользователей.

-u

`set-user-id` (`suid`) флаг для файла установлен

Установленный флаг `suid` приводит к изменению привилегий запущенного процесса на привилегии владельца исполняемого файла. Исполняемые файлы, владельцем которых является *root*, с установленным флагом *set-user-id* запускаются с привилегиями *root*, даже если их запускает обычный пользователь. [1] Это может оказаться полезным для некоторых программ (таких как *pppd* и *cdrecord*), которые осуществляют доступ к аппаратной части компьютера. В случае отсутствия флага *suid*, программы не смогут быть запущены рядовым пользователем, не обладающим привилегиями *root*.

```
-rwsr-xr-t    1 root    178236 Oct  2  2000 /usr/sbin/pppd
```

Файл с установленным флагом *suid* отображается с включенным флагом *s* в поле прав доступа.

-k

флаг *sticky bit* (бит фиксации) установлен

Общеизвестно, что флаг "sticky bit" -- это специальный тип прав доступа к файлам. Программы с установленным флагом "sticky bit" остаются в системном кэше после своего завершения, обеспечивая тем самым более быстрый запуск программы. [2] Если флаг установлен для каталога, то это приводит к ограничению прав на запись. Установленный флаг "sticky bit" отображается в виде символа *t* в поле прав доступа.

```
drwxrwxrwt    7 root    1024 May 19 21:26 tmp/
```

Если пользователь не является владельцем каталога, с установленным "sticky bit", но имеет право на запись в каталог, то он может удалять только те файлы в каталоге, владельцем которых он является. Это предотвращает удаление и перезапись "чужих" файлов в общедоступных каталогах, таких как /tmp.

-O

вы являетесь владельцем файла

-G

вы принадлежите к той же группе, что и файл

-N

файл был модифицирован с момента последнего чтения

f1 -nt f2

файл *f1* более новый, чем *f2*

f1 -ot f2

файл *f1* более старый, чем *f2*

f1 -ef f2

файлы *f1* и *f2* являются "жесткими" ссылками на один и тот же файл

!

"НЕ" -- логическое отрицание (инверсия) результатов всех вышеприведенных проверок (возвращается true если условие отсутствует).

Пример 7-4. Проверка "битых" ссылок

```
#!/bin/bash
# broken-link.sh
# Автор Lee Bigelow <ligelowbee@yahoo.com>
# Используется с его разрешения.

#Сценарий поиска "битых" ссылок и их вывод в "окавыченном" виде
#таким образом они могут передаваться утилите xargs для дальнейшей обработки :)
#например. broken-link.sh /somedir /someotherdir|xargs rm
#
#На всякий случай приведу лучший метод:
#
#find "somedir" -type l -print0|\
#xargs -r0 file|\
#grep "broken symbolic"|
#sed -e 's/^\ |: *broken symbolic.*$/"/g'
#
#но это не чисто BASH-евский метод, а теперь сам сценарий.
#Внимание! будьте осторожны с файловой системой /ргос и циклическими ссылками!
#####

#Если скрипт не получает входных аргументов,
#то каталогом поиска является текущая директория
#В противном случае, каталог поиска задается из командной строки
#####
[ $# -eq 0 ] && directorys='pwd' || directorys=$@

#Функция linkchk проверяет каталог поиска
#на наличие в нем ссылок на несуществующие файлы, и выводит их имена.
#Если анализируемый файл является каталогом,
#то он передается функции linkcheck рекурсивно.
#####
linkchk () {
    for element in $1/*; do
        [ -h "$element" -a ! -e "$element" ] && echo "\"$element\""
        [ -d "$element" ] && linkchk $element
        # Само собой, '-h' проверяет символические ссылки, '-d' -- каталоги.
    done
}

#Вызов функции linkchk для каждого аргумента командной строки,
```

```
#если он является каталогом.  Иначе выводится сообщение об ошибке
#и информация о порядке пользования скриптом.
#####
for directory in $directorys; do
    if [ -d $directory ]
        then linkchk $directory
        else
            echo "$directory не является каталогом"
            echo "Порядок использования: $0 dir1 dir2 ..."
        fi
done

exit 0
```

[Пример 28-1](#), [Пример 10-7](#), [Пример 10-3](#), [Пример 28-3](#) и [Пример А-2](#) так же иллюстрируют операции проверки файлов.

Примечания

- [1] С флагом *suid*, на двоичных исполняемых файлах, надо быть очень осторожным, поскольку это может быть небезопасным. Установка флага *suid* на файлы-сценарии не имеет никакого эффекта.
- [2] В современных UNIX-системах, "sticky bit" больше не используется для файлов, только для каталогов.

[Назад](#)[К началу](#)[Вперед](#)[Проверка условий](#)[Наверх](#)[Операции сравнения](#)

Партнёры:



При поддержке
inferno solutions*

Хостинг:



Hoster.ru
хостинг провайдер

[Закладки на сайте](#)
[Проследить за страницей](#)

Created 1996-2024 by [Maxim Chirkov](#)
[Добавить](#), [Поддержать](#), [Вебмастеру](#)