Оболочка, или шелл (shell) — это программа, в нашем случае названная «bash», что является сокращением от Bourne Again Shell. Оболочка принимает ваши команды и передаёт их операционной системе. Для взаимодействия с системой используются терминалы, такие как [gnome-terminal](https://help.gnome.org/users/gnome-terminal/stable/" \t "_blank), ~~[eterm](http://www.eterm.org/" \t "_blank)~~, nxterm и т. п.

Навигация

В Linux файлы и каталоги имеют иерархическую организацию, то есть существует некий начальный каталог, называемый корневым. В нём содержатся файлы и подкаталоги, которые в свою очереди содержат файлы и свои подкаталоги.

pwd

Команда pwd, сокращение от *print working directory*, отображает текущее местоположение в структуре каталогов.

cd

Команда cd позволяет перейти в новый каталог.

| **Синтаксис** | **Объяснение** |
| --- | --- |
| cd | Перемещение в домашний каталог |
| cd ~ | Перемещение в домашний каталог |
| cd .. | Перемещение на один уровень выше |
| cd - | Перемещение в предыдущий каталог |
| cd Directory1 | Перемещение в каталог Directory1 |
| cd Directory1/Directory2 | Перемещение в каталог Directory2 по указанному пути |

mkdir

Команда mkdir создаёт новый каталог в текущем каталоге.

Основные команды

man

Команда man отображает руководства по командам. Например, следующая команда выдаст всю информацию о команде cat:

$ man cat

cat

Команда cat считывает файл, переданный как аргумент, и выводит его содержимое по стандартному каналу вывода. Передача нескольких файлов в виде аргумента приведёт к выводу конкатенированного содержимого всех файлов.

echo

Команда echo выводит свои аргументы по стандартному каналу вывода.

$ echo Hello World

Hello World

Если вызвать echo без аргументов, будет выведена пустая строка.

head

Команда head читает первые 10 строк любого переданного текста и выводит их по стандартному каналу. Число выводимых строк можно изменить:

$ head -50 test.txt

tail

Команда tail работает аналогично команде head, но читает строки с конца:

$ tail -50 test.txt

Также можно просматривать добавляемые к файлу строки в режиме реального времени при помощи флага -f:

$ tail -f test.txt

less

Команда less позволяет перемещаться по переданному файлу или куску текста, причём в обоих направлениях.

$ less test.txt

$ ps aux | less

Подробнее о назначении символа | будет рассказано ниже в разделе команды history.

| **Обычные сочетания клавиш** | **Описание** |
| --- | --- |
| G | Перемещает в конец файла |
| g | Перемещает в начало файла |
| :50 | Перемещает на 50 строку файла |
| q | Выход из less |
| /searchterm | Поиск строки, совпадающей с ‘searchterm’, ниже текущей строки |
| / | Перемещает на следующий подходящий результат поиска |
| ?searchterm | Поиск строки, совпадающей с ‘searchterm’, выше текущей строки |
| ? | Перемещает на следующий подходящий результат поиска |
| up | Перемещает на одну строку выше |
| down | Перемещает на одну строку ниже |
| pageup | Перемещает на одну страницу выше |
| pagedown | Перемещает на одну страницу ниже |

true

Команда true всегда возвращает ноль в качестве выходного статуса для индикации успеха.

false

Команда false всегда возвращает не-ноль в качестве выходного статуса для индикации неудачи.

$?

$? — это переменная, которая содержит выходной статус последней запущенной команды. Под статусом обычно понимается [код возврата](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4_%D0%B2%D0%BE%D0%B7%D0%B2%D1%80%D0%B0%D1%82%D0%B0) программы. 0 означает успешное выполнение программы, любое значение большее 0 отражает тот факт, что в процессе выполнения возникли некоторые ошибки. Кстати, именно поэтому в bash истинной (true) считается 0, а все, что не 0 — ложью (false):

$ true

$ echo $?

0

$ false

$ echo $?

1

grep

Команда grep занимается поиском переданной строки в указанном файле:

$ cat users.txt

user:student password:123

user:teacher password:321

$ grep 'student` file1.txt

user:student password:123

grep также может принимать несколько файлов и регулярных выражений для уточнения формата текста.

| **Обычные флаги** | **Описание** |
| --- | --- |
| -i | Отключение чувствительности к регистру |
| -r | Рекурсивный поиск по директориям |
| -w | Поиск только целых слов |
| -c | Вывод количества найденных элементов |
| -n | Вывод всей строки, содержащей запрос |
| -v | Вывод инвертированного совпадения |

sed

Команда sed — это потоковый редактор, преобразующий входные текстовые данные. Обычно её используют для замены выражений так: s/regexp/replacement/g. Например, следующий код заменит все слова «Hello» на «Hi»:

$ cat test.txt

Hello World

$ sed 's/Hello/Hi/g' test.txt

Hi World

history

Команда history выводит историю командной строки. Обычно её используют вместе с командой grep для поиска конкретной команды. Например, следующий код найдёт все команды, содержащие строку g++:

$ history | grep g++

155 g++ file1.txt

159 g++ file2.txt

Здесь также используется символ | — это так называемый [конвейер](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D0%BD%D0%B0%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B2%D0%B2%D0%BE%D0%B4%D0%B0-%D0%B2%D1%8B%D0%B2%D0%BE%D0%B4%D0%B0) (pipe). Благодаря ему можно перенаправлять вывод одной команды на вход другой — таким образом в примере выше вся история, которая в обычном режиме выводится командой historyпрямо в вывод терминала, будет перенаправлена в grep в качестве входных данных. Мы не увидим вывода команды history, но увидим вывод команды grep.

Это может быть довольно сложно для понимания без практики, поэтому поэкспериментируйте самостоятельно, например с командами ls, history, ps(описана ниже), перенаправляя их вывод в grep, sed или less, например.

export

Команда export устанавливает переменные окружения для передачи дочерним процессам. Например, так можно передать переменную name со значением student:

$ export name=student

ps

Команда ps выводит информацию о запущенных процессах.

$ ps

PID TTY TIME CMD

35346 pts/2 00:00:00 bash

Выводится четыре элемента:

* ID процесса (PID),
* тип терминала (TTY),
* время работы процесса (TIME),
* имя команды, запустившей процесс (CMD).

awk

Команда awk находит и заменяет текст в файлах по заданному шаблону: awk 'pattern {action}' test.txt

wget

Команда wget скачивает файлы из Сети и помещает их в текущий каталог.

$ wget https://github.com/mikeizbicki/ucr-cs100

nc

Команда nc — это утилита для отладки сети.

ping

Команда ping тестирует сетевое подключение.

$ ping google.com

PING google.com (74.125.224.34) 56(84) bytes of data.

64 bytes from lax17s01-in-f2.1e100.net (74.125.224.34): icmp\_req=1 ttl=57 time=7.82 ms

--- google.com ping statistics ---

1 packets transmitted, 1 received, 0% packet loss, time 8ms

rtt min/avg/max/mdev = 7.794/8.422/10.792/0.699 ms

Статистика в конце показывает количество подключений, совершённых до завершения команды, и время их выполнения.

git

Git — это популярная система контроля версий.

Переменные окружения

Переменные окружения — это именованные переменные, содержащие значения, используемые одним или несколькими приложениями.

Переменная PATH содержит список каталогов, в которых система ищет исполняемые файлы.

Переменная HOME содержит путь к домашнему каталогу текущего пользователя.

Коннекторы

Коннекторы позволяют запускать несколько команд одновременно.

| **Коннектор** | **Описание** |
| --- | --- |
| && | Первая команда исполняется всегда, вторая — только в случае успешного завершения первой |
| || | Первая команда исполняется всегда, вторая — только в случае неудачного завершения первой |
| ; | Команды исполняются всегда |

$ true && echo Hello

Hello

$ false || echo Hello

Hello

$ echo Hello ; ls

Hello

test.txt file1.txt file2.txt

Конвейеры

Конвейеры, или пайпы, позволяют соединять входные и выходные каналы различных команд. В следующем примере вывод команды ls будет передан в head , и в результате будет напечатано лишь 10 первых элементов.

$ ls -l | head

Перенаправление ввода/вывода

Перенаправление вывода

Для стандартного перенаправления вывода используются символы > и >>.

Например, этот код передаст вывод ls в файл, а не на экран:

$ ls > files.txt

$ cat files.txt

file1.cpp sample.txt

Если файл не существует, он создаётся, а если существует, то перезаписывается. Во избежание перезаписи стоит использовать команду >> — она дописывает данные в конец файла.

Перенаправление ввода

Для стандартного перенаправления вывода используется символ <. В следующем примере sort берет входные данные из файла, а не с клавиатуры:

$ cat files.txt

c

b

$ sort < files.txt

b

c

Команда sort выводит содержимое файла на экран, поскольку мы не перенаправили выход. Это можно сделать так:

$ sort < files.txt > files\_sorted.txt

Продвинутое перенаправление

Добавление & к > приводит к перенаправлению как стандартного потока выхода, так и потока ошибок. Например, файл test.cpp выведет строку stdout в cout и строку stderr в cerr.

$ g++ test.cpp

$ ./a.out >& test.txt

$ cat test.txt

stdout

stderr

Если вы хотите вывести конкретный файловый дескриптор, вы можете приписать его номер к >.

| **Имя** | **Дескриптор** | **Описание** |
| --- | --- | --- |
| stdin | 0 | Стандартный поток ввода |
| stdout | 1 | Стандартный поток вывода |
| stderr | 2 | Стандартный поток вывода ошибок |

Например, для перенаправления stderr в test.txt нужно сделать следующее:

$ g++ test.cpp

$ ./a.out 2> test.txt

stdout

$ cat test.txt

stderr

Права доступа

Команда ls -l выводит много информации о права доступа к каждому файлу:

$ ls -l test.txt

-rw-rw-r-- 1 user group 1097374 January 26 2:48 test.txt

| **Вывод в примере** | **Описание / возможные выводы** |
| --- | --- |
| — | Тип файла: - файл d каталог |
| rw- | Права доступа владельца файла |
| rw- | Права доступа членов группы-владельца файла |
| r– | Права доступа прочих пользователей |
| user | Имя владельца файла |
| group | Имя группы-владельца файла |

chmod

Команда chmod изменяет права доступа файла. Вот типичные сочетания флагов для изменения прав конкретных пользователей:

| **Буква** | **Пользователь** |
| --- | --- |
| u | Владелец |
| g | Член группы |
| o | Прочие пользователи |
| a | Все пользователи |

Вы можете вызвать chmod с описанием действий над конкретным файлом. Символ - обозначает удаление прав, символ + — добавление. Следующий пример сделает файл доступным для чтения и записи владельцу и группе:

$ chmod ug+rw test.txt

$ ls -l test.txt

-rw-rw---- 1 user group 1097374 January 26 2:48 test.txt

Кроме того, chmod можно использовать с восьмеричными числами, где 1 — это наличие прав, а 0 — отсутствие:

rwx = 111 = 7

rw- = 110 = 6

r-x = 101 = 5

r-- = 100 = 4

Следующая команда сработает так же, как и предыдущая:

$ chmod 660 test.txtСочетания клавиш

| **Сочетание** | **Описание** |
| --- | --- |
| CTRL-A | Перемещение курсора в начало строки |
| CTRL-E | Перемещение курсора в конец строки |
| CTRL-R | Поиск по истории |
| CTRL-W | Вырезать последнее слово |
| CTRL-U | Вырезать всё до курсора |
| CTRL-K | Вырезать всё после курсора |
| CTRL-Y | Вернуть последнюю вырезанную строку |
| CTRL-\_ | Отмена |
| CTRL-L | Очистка экрана терминала |