



# Administrator Linux.Basic

**Bash.**

**Написание простых скриптов**



Проверить, идет ли запись

**Меня хорошо видно  
&& слышно?**



# Преподаватель



## Лавлинский Николай

Технический директор «Метод Лаб»

Более 15 лет в веб-разработке

Преподавал в ВУЗе более 10 лет

Более 4 лет в онлайн-образовании

Специализация: оптимизация производительности, ускорение сайтов и веб-приложений

[https://t.me/methodlab\\_tg](https://t.me/methodlab_tg)

<https://www.methodlab.ru/>

<https://www.youtube.com/c/NickLavlinsky>

[https://www.youtube.com/@site\\_support](https://www.youtube.com/@site_support)

<https://vk.com/nick.lavlinsky>

# Правила вебинара



Активно  
участвуем



Off-topic обсуждаем  
в Телеграм-чате



Задаем вопрос  
в чат или ГОЛОСОМ



Вопросы вижу в чате,  
могу ответить не сразу

## Условные обозначения



Индивидуально



Время, необходимое  
на активность



Пишем в чат



Говорим голосом



Документ



Ответьте себе или  
задайте вопрос

# Карта курса

Введение  
в Linux

Основные сервисы:  
Nginx, MySQL и Git

Сети в Linux

Логирование

Мониторинг

Astra Linux

Собственный проект

# Маршрут вебинара



Что такое bash?

Переменные

Подстановки

Условия

Циклы

# Цели вебинара

После занятия вы сможете

1. Более эффективно работать в терминале
2. Писать однострочные bash-скрипты
3. Создавать и читать сложные bash-скрипты

# Смысл

## Зачем вам это уметь

1. Заменять одним однострочником множество команд
2. Читать и модифицировать готовые bash-скрипты
3. Автоматизировать рутинные задачи
4. Решать проблемы вызова команд





# Что такое bash?



# Что такое bash?

- Bourne-again shell
- Интерактивная оболочка
- Язык программирования
- Стандартный вариант
- Другие варианты
  - zsh
  - csh
  - ksh
  - sh



# Bash-скрипт

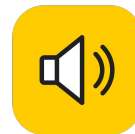
- Текстовый файл
- Состоит из команд и операторов bash
- Расширение файла — обычно нет, можно .sh
- Разделитель команд — новая строка или точка с запятой
- Не компилируется в машинный код
- Исполняется интерпретатором - bash
- **Как система определяет тип скрипта?**



# Что такое команды?

```
Commands
```

```
ls  
bash  
find  
cd ..  
cat  
ll  
type
```



# Типы команд

Command types



```
$ type -a cd  
cd is a shell builtin
```

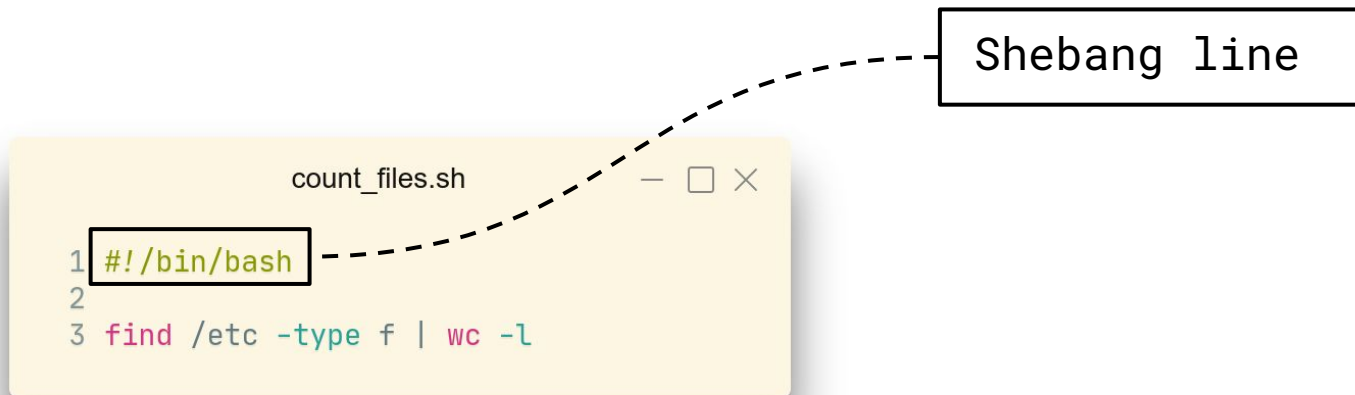
```
$ type -a fgrep  
fgrep is /bin/fgrep
```

```
$ type -a ls  
ls is aliased to 'ls --color=auto'  
ls is /bin/ls
```

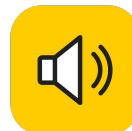
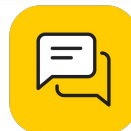
```
$ type -a for  
for is a shell keyword
```

```
$ which passwd  
/usr/bin/passwd
```

# Запускаем простой скрипт



Как запустить скрипт?



# Методы запуска

Terminal

— □ ×

```
# 1 - без прав
bash count_files.sh

# 2 - с правом на исполнение
chmod +x count_files.sh

# Относительный путь
./count_files.sh

# Абсолютный путь
/home/db/count_files.sh

# 3 - из директории для исполняемых файлов
echo $PATH
/usr/local/bin
cp ./count_files.sh /usr/local/bin/
```

# Позиционные параметры

- `$1` - первый параметр
- `$2` - второй
- `$#` - количество
- `${10}` - 10-й
- `${11}` - 11-й



# Использование параметров

Parameters



```
# count_files.sh  
find $1 -type f | wc -l  
  
bash count_files.sh /usr  
bash count_files.sh /etc  
bash count_files.sh
```

# Переменные

# Переменные

- Временные значения
- Видны в рамках одного процесса (скрипта)
- Пользовательские переменные `$var`
- Системные переменные (константы) `$PATH`
- Могут наследоваться через `export`

# Использование переменных

Variables

— □ ×

```
training=Python  
TRAINING=Linux
```

```
echo $TRAINING  
echo $training  
echo ${TRAINING}  
echo ${training}
```

```
# count_files.sh  
directory=$1  
find $directory -type f | wc -l
```

# А так можно?

Variables names



```
1var=2
```

```
var =2
```

```
var = 2
```

```
VaR=2
```

```
Var1=2
```

```
var_=2
```

```
var_test=2
```

```
перемен1=2
```



# Что это такое?

Variables



```
$(command)
```

```
output=$(cat /etc/passwd)
```

```
output=`cat /etc/passwd`
```



# Результат команды в переменную

Variables exec 2



```
# count_files.sh
directory=$1
num_of_files=$(find $directory -type f | wc -l)
echo $num_of_files
```

# Запрос ввода пользователя, сохранение в переменную

User input



```
1 #!/bin/bash
2
3 echo "Enter the filename:"
4 read filename
5
6 echo $filename
```



# Подстановки

# Подстановки (expansions)

Expansions

— □ ×

```
echo {0..10}
```

```
mkdir test{a..e}
```

```
echo {{a..e},z}
```

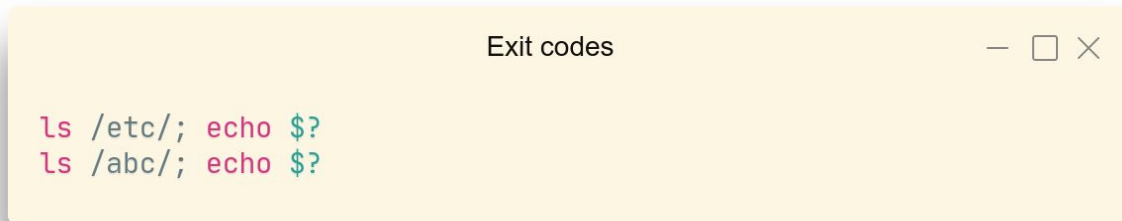
```
mkdir test{001..100}
```

```
echo ~
```

```
echo ~+
```

```
echo ~-
```

# Код возврата (exit code)



```
Exit codes
```

```
ls /etc/; echo $?  
ls /abc/; echo $?
```

- Выставляется в результате любой команды
- 0 — команда выполнена успешно
- 1..255 — команда выполнялась с ошибкой (код ошибки)
- Код возврата доступен через переменную \$?
- Позволяет организовывать условия в программе

# Условия

# Условия (if)

if and test



```
if [ -f $filename ]  
then  
    echo "True!"  
else  
    echo "False!"  
fi
```

```
type -a test [ [  
test -f /etc/ ; echo $? # существует ли файл?  
test -d /etc/ ; echo $? # существует ли директория?
```

# Команда test и [ или [[

- Если условие успешно, возвращает `exit code 0`, иначе 1
- `==` — равенство.
- `!=` — неравенство.
- `-lt` — меньше.
- `-gt` — больше.
- `-lte` — меньше или равно.
- `-gte` — больше или равно.
- `-f` — файл.
- `-d` — директория
- `help test`; `help [`; `help [[`
- `man test` # для внешней команды

# Практика

# Задача

```
Conditional exit
```

```
1 #!/bin/bash
2
3 # count_files.sh
4
5 directory="$1"
6 find "$directory" -type f | wc -l
```

Написать скрипт который проверяет, что пользователь передал аргумент.  
Если аргумента нет — выдать сообщение об ошибке и завершить программу



## Conditional exit



```
1 # Вариант 1
2 if [ -z "$directory" ]
3 then
4     echo Please specify directory
5     exit 1
6 fi
7
8 # Вариант 2
9 if [ $# -eq 0 ]
10 then
11     echo Please specify directory
12     exit 1
13 fi
14
15 # Вариант 3
16 directory=$1
17 if [ -n "$directory" ]
18 then
19     find "$directory" -type f | wc -l
20 else
21     echo Please specify directory
22     exit 1
23 fi
```

# Циклы

# Цикл for

For cycles

— □ ×

```
1 #!/bin/bash
2
3 for h in {01..24}
4 do
5     echo $h
6 done
7
8 for i in *
9 do
10    echo $i
11 done
12
13 for (( c=1; c<=5; c++ ))
14 do
15     echo "Попытка номер $c"
16 done
```

# Цикл while

While cycle



```
1 #!/bin/bash
2
3 c=10
4
5 while [ $c -ge 0 ]
6 do
7     echo "Test"
8     let "c = c - 1"
9 done
```

# Вопросы?



Ставим "+",  
если вопросы есть



Ставим "-",  
если вопросов нет

# Что мы изучили?

## Подведем итоги

1. Разобрались, что такое bash
2. Изучили работу с переменными
3. Использовали условия в скриптах
4. Посмотрели на два вида циклов: for и while



# Домашнее задание 1

Необходимо написать скрипт, который очищает указанную папку от временных файлов.

1. Первым параметром получает директорию.
2. Проверяет, что указанная директория существует. Если нет – пишет ошибку и завершается с кодом 2.
3. Удаляет все файлы \*.bak, \*.backup, \*.tmp. Для каждого типа выводит сообщение об удалении или об отсутствии таких файлов.
4. Прислать исходный код скрипта в кодировке UTF-8 в текстовом файле.



# Домашнее задание 2\* (дополнительно)

Необходимо написать скрипт, который управляет настройками DNS в `/etc/resolv.conf`.  
Прислать исходный код скрипта в кодировке UTF-8 в текстовом файле.

1. Получает на вход список из двух DNS-серверов. Если на входе меньше или больше — ошибка.
2. Проверяет текущую конфигурацию в `/etc/resolv.conf` (соответствует ли настройка).
3. Проверяет, есть ли права у пользователя на редактирование.
4. Если конфигурация не совпадает, то выводит об этом сообщение и меняет настройки в файле из параметров.
5. У скрипта должен быть режим справочника (помощи). В справке показывается синтаксис и описывается использование скрипта.





# Рефлексия

# Цели вебинара

## Проверка достижения целей

1. Более эффективно работать в терминале
2. Писать однострочные bash-скрипты
3. Создавать и читать сложные bash-скрипты



# Вопросы для проверки

## Вопросы для проверки

1. Какие бывают типы переменных?
2. Какие типы команд вы знаете и как их узнавать?
3. Что делает команда test?



# Рефлексия



Что было самым полезным на занятии?



Как будете применять на практике то, что узнали на вебинаре?

# Следующий вебинар



## Конфигурирование web-сервера (apache, nginx, балансировка nginx)



Ссылка на  
вебинар будет  
в ЛК за 15 минут



Материалы  
к занятию в ЛК —  
можно изучать



Обязательный  
материал обозначен  
красной лентой



**Заполните, пожалуйста,  
опрос о занятии  
по ссылке в чате**

Спасибо за внимание!

# Приходите на следующие вебинары



**Лавлинский Николай**

Технический директор “Метод Лаб”

<https://www.methodlab.ru/>

<https://www.youtube.com/c/NickLavlinsky>

[https://www.youtube.com/@site\\_support](https://www.youtube.com/@site_support)

<https://vk.com/nick.lavlinsky>

