Пайпы, shell-глоббинг

Одна из важнейших частей работы инженера эксплуатации — автоматизация задач, то есть делегирование ручных действий скриптам, коду или специальным программам.

С этого момента мы начнём изучать написание скриптов с использованием шелла bash. Это самый простой пример автоматизации действий, но достаточно мощный, чтобы помогать нам в работе.

Скрипт — простейший случай автоматизации наших действий. Как и любая программа, скрипт — это последовательность действий, описанная специальным языком, который какой-то интерпретатор может понять и выполнить.
В случае bash-скриптов — командная строка.

Шаблон поиска (шелл-глоббинг)

В названиях файлов картинок есть та самая общая часть, которая позволит нам воспользоваться шаблоном, — одинаковое расширение, .jpeg.
Шаблон для поиска таких файлов будет выглядеть таким образом: *.jpeg, то есть любой текст, а затем .jpeg.

Теперь нужно передать наш шаблон команде, которая их понимает. Например, подойдёт ls:

```
ls -la *.jpeg
```

* — это шаблон, с которым совпадает любая строка символов. Чтобы отобразить все файлы в директории, мы можем воспользоваться им же:

```
ls *
```

Найти все офисные документы, .docx, pptx, xslx

У всех этих расширений есть кое-что общее. Во-первых, они состоят из четырёх букв. Во-вторых, у них на конце **x**.

Воспользуемся вторым символом шаблонов: ?. Знак вопроса совпадает с любым одиночным символом, так что наш шаблон будет выглядеть так:

```
*.???x
```

На компьютере есть папка, в которой содержатся разные документы. Давайте проверим:

```
ls -la *.???x
```

Конвейер (Pipe).
В русской терминологии используется или английское слово pipe, или «конвейер».

В случае с UNIX конвейер работает так: результат работы одной программы мы можем отправить другой программе на обработку.
Это может понадобиться, когда одна программа открывает файл, а другая получает из него только нужные данные. Или с помощью одной программы мы создаём некий список, с помощью другой — сортируем его в алфавитном порядке.

Узнать, сколько файлов лежит в директории
Мы можем получить список файлов:

```
ls /var/log
```

И просто посчитать в уме или ставя заметки на листочке, но это неэффективно.

Понадобится команда, чтобы вывести все файлы в директории и посчитать количество получившихся строк.
Сначала нужно найти файлы в директории при помощи команды find:

```
find . -type f
```

Чтобы передать то, что выведет эта команда другой, мы воспользуемся символом пайпа |.
Чтобы посчитать файлы, мы используем команду wc.
wc умеет считать объекты — строки, слова, символы. Так как find отдаёт нам имена файлов по одному на строку, мы можем посчитать их при помощи wc -l.
Полностью команда будет выглядеть так:

```
find . -type f . | wc -l
```

При помощи одной команды мы выяснили, сколько в нашей директории файлов.

Команда *mv – move* служит для сортировки папки, чтобы переносить файлы из одной директории в другую.

```
mv /home/victoria/file /home/devops/
```

Представим, что для директорий эта команда не работает.
Пусть в нашей домашней директории будет две папки — pine и forest. Мы хотим перенести папку pine внутрь папки forest.

Для написания скрипта откроем текстовый редактор:
1. Нужно скопировать одну директорию в другую. Как мы уже знаем, для копирования мы используем команду *cp*. А для того, чтобы скопировать не файл, а директорию, используем ключ -r (как и в случае с командой rm, кстати).
2. Нужно удалить директорию, для этого есть команда rm. К ней добавим ключ -r, чтобы она удалила директорию и -f и не спрашивала подтверждения.

Представим, что у нас есть две директории рядом — pine и forest. Напишем сначала скрипт для частного случая:

```
cp -r pine forest
```

```
rm -rf forest
```

Но этот скрипт сработает только для двух конкретных папок, а команде mv мы можем передавать аргументы — имя файла и папку, куда хотим переместить.

Чтобы получить аналогичное поведение, мы можем использовать специальные параметры. Они составляются так: $ + номер. При помощи них мы изнутри скрипта сможем получить значение, которое ввели в командной строке после самого имени скрипта. $1 — первое, $2 — второе и так далее.

Сделаем по аналогии с командой mv и первым аргументом будем ожидать путь, по которому находится наш каталог, который мы переносим, а вторым — путь, куда мы хотим перенести каталог.

```
cp -r $1 $2
rm -rf $1
```

Далее нужно указать:

- что у нас не просто текстовый файл, а скрипт;
- кто именно должен его исполнять.

Для того чтобы наш скрипт заработал, нам нужно объяснить компьютеру, что от него требуется. Нужен своего рода переводчик. Или, если точнее, интерпретатор. Здесь интерпретатором будет bash.

```
/bin/bash
cp -r $1 $2
rm -rf $1
```

А чтобы указать шеллу на то, что это интерпретатор, а не просто текстовая строка, воспользуемся специальным набором символов #!

```
#!/bin/bash
cp -r $1 $2
rm -rf $1
```

Кстати, #! называется шебанг, или хэшбэнг.

Сохраним скрипт в файле под названием mvdir.sh.

Протестируем скрипт

Когда запускаем команду в графической среде, мы просто кликаем на иконке, но в консоли для этого нужно написать имя программы и необходимые ей аргументы.

Когда мы запускаем скрипт, который не находится там, где Linux ожидает увидеть программы (/bin /usr/bin и так далее), нам нужно явно указать, что скрипт находится в нашей текущей директории.

```
./mvdir.sh pine forest
```

Ошибка!

Даём право на исполнение, execute (x):

```
chmod +x mvdir.sh
./mvdir.sh pine forest
ls forest
```

Первый баш-скрипт написан и работает!

Вернёмся к сортировке файлов в папке

Давайте напишем пример, который позволит нам переносить все картинки из папки Downloads в папку Pictures.
Для этой задачи нам хватит того, что умеет утилита find.

У неё есть замечательная возможность, -exec, она позволяет проводить над найденными файлами какие-нибудь действия. В данном случае нам потребуется их перенести.

Начнём формулировать команду.
Сначала нам потребуется найти все файлы с расширением jpeg.

```
find /home/victoria/Downloads -name "*.jpeg"
```

Затем воспользуемся возможностью передать действие через exec.

```
-exec mv {} /home/victoria/Pictures \;
```

Вместо пустых фигурных скобок у нас будет подставляться список файлов, найденный по условию, конструкция `\;` закрывает нашу команду. Она верна только для похожих случаев, а не для любой команды и, к сожалению, не для любого случая использования exec. Позже мы вернёмся к тому, что она означает, а пока ею просто можно пользоваться.
Итак, мы обзавелись однострочником, который позволяет нам собирать все картинки и переносить их в папку для картинок.