Если вы видите что-то необычное, просто сообщите мне.

Linux\Bash Работа в командной строке.

Права доступа к файлам

У файла есть три группы владельцев:

- User(пользователь)
- Group(группа)
- Other(все другие)

Каждой группе назначается права доступа:

- Read(Чтение)
- Write(Запись)
- Execute(Выполнение)

Рассмтрим пример Выолните в командной строке:

```
ls -l
```

В ответ мы получим вывод следующего вида:

```
-rw-r--r--.  1 user user         478 сен  6 16:42  index.html
-rwxr-xr-x.  1 user user        1244 июл 30 10:37  modules
drwxr-xr-x.  5 user user        4096 сен 10 15:58  Pictures
drwxr-xr-x.  2 user user        4096 янв 24  2021  Public
drwxr-xr-x. 13 user user        4096 авг 17 09:12  Repos
```

Разрберем запись:

```
drwxr-xr--
```

Нотация имеет группы свойств:

- `d` или `-` укзание того, что перед нами файл или папка.
- `rwx` - позволяет пользователю читать(`r`), писать(`w`), исполнять(`x`) этот файл
- `r-x` - позволяет группе пользователей читать(`r`), но не писать в него (`-`), испольнять(`x`)
- `r-x` - позволяет всем остальным читать(`r`), но не писать в него (`-`), не испольнять(`-`)

`---` - подразумевает что у владельца, группы или остальных нет никаких разрешений вообще.

CHMOD

Довольно часто для смены прав доступа используется команда `chmod`.

```
chmod permissions filenam
```

Используя табилцу ниже файлу или папке можно дать различные права:

Число	Тип разрешения	Символ
0	Отсутствует разрешение на	---
1	Выполнение	--x
2	Запись	-w-
3	Выполнение+ Запись	-wx
4	Чтение	r--
5	Чтение+Выполнение	r-x
6	Чтение+Запись	rw-
7	Чтение+Запись+ВЫполнение	rwx

CHOWN

Команда для изменения владельца файла\каталога

```
chown user:group filename
```

Возьмем пример выше

```
-rwxr-xr-x.  1 user user        1244 июл 30 10:37  modules
```

Если мы воспользуемся командой chown:

```
chown test:group modules
```

То в результате `user` должен смениться на `test`, и `user` на `group`, и при проверке прав мы увидим следующие изменения:

```
-rwxr-xr-x.  1 test group        1244 июл 30 10:37  modules
```

Перенаправление ввода

">"

Этот символ используется для перенаправления вывода(stdout)

Пример:

```
la -al > file.txt
```

В этом случае всё что выведется командой `la -al` будет записано в `file.txt`, при этом на экране ничего не отобразиться.

Можно рассмотреть следующий пример:

```
cat music.mp3 > /dev/audio
```

Команда `cat` прочитает файл `music.mp3` и вместо того чтобы вывести его в терминал отправит в устройство `/dev/audio`, если у вас в системе есть такое устройство то вы услышите этот файл из колонок.

"<"

Этот символ используется для перенаправления ввода(stdin)

Пример:

```
mail -s "subject" to-address < Filename
```

Команда прикрепит `Filename` к письму и отправит его по адресу `to-address`.

Перенаправление ошибок

Всякий раз когда выполняется программа открыто 3 файла:

- standard input -- FD0
- standard output -- FD1
- standard error -- FD2

FD - файловый дескриптор. В Linux / Unix все является файлом. Обычный файл, каталоги и даже устройства являются файлами. Каждый файл имеет связанный номер, называемый дескриптором файла (FD).

примеры использования:

1. Вывод ошибок программы `myprogram` перенаправляется в errorfile. В этом случае ошибки не отображаются при выполнении команды `myprogram`.

```
myprogram 2>errorsfile
```

2. С помощью `find` мы ищем в текущем каталоге `.` файй начинающийся с `my`. При этом все ошибки описка отправляются в файл `error.log`

```
find . -name 'my*' 2>error.log
```

3. Команда выводит список файлов в двух директориях `Documents` и `ABC`, в свою очередь `2>&1` переводи вывод ошибок в стандартный вывод. То есть `stderr` перенаправляется в `stdout`. И весь вывод записывается в файл dirlist

```
ls Documents ABC> dirlist 2>&1
```

Pipeline

Pipeline - комбинация команд передающая аргументы друг другу после выполнения. Т.е. вывод команды является вводом следующей команды

Например, ниже мы передаем вывод команды `cat` на вход команды `less`:

```
cat filename | less
```

grep

Создадим файл `example.txt`, внесем в него какие-либо строки и выведем его:

```
cat example.txt
```

Команда `grep` выведет искомые данные. Синтаксис использования: `grep ИСКОМОЕ_ВЫРАЖЕНИЕ`:

```
cat example.txt | grep dog
```

Команда `grep` пройдется по выводу команды `cat` и отразит только строки содержащие `dog`.

Ключи команды `grep`:

ключь	функция
-v	Показывает все строки, которые не соответствуют искомой строке
-c	Отображает только количество совпадающих строк
-n	Показывает совпадающую строку и ее номер
-l	Показывает только имя файла со строкой

sort

Сортирует содержимое в алфавитном порядке

Команда `sort` выведет искомые данные. Синтаксис использования: `sort ФАЙЛ`.

```
sort example.txt
```

Ниже приведен вариант использования `sort` в комбинации с командой `grep` в pipeline:

```
cat example.txt | grep -v a | sort -r
```

`grep` выведет все строки не содержащие букву `a`, а затем отсортирует их в обратном алфавитном порядке.

Регулярные выражения

Регулярные выражения — это специальные символы, которые помогают искать данные, соответствующие сложным шаблонам. Регулярные выражения сокращаются до ‘regexp’ или ‘regex’.

Основные Регулярные выражения

Символ	Описания
,	заменяет любой символ
^	соответствует началу строки

Символ	Описания
$	соответствует концу строки
*	соответствует нулю или более раз предыдущего символа
\	Представляют специальные символы
()	Группы регулярных выражений
?	Соответствует ровно одному символу

Интервал в регулярных выражениях

Выражение	Описание
{n}	Соответствует предыдущему символу, появляющемуся `n` раз точно
{n,m}	Соответствует предыдущему символу, появляющемуся `n` раз, но не более чем `m`
{n,}	Соответствует предыдущему символу, только если он появляется `n` раз или более

Примеры использования:

Создадим файл `example.txt` и внесем в него какой-то текст

Чтобы посмотреть содержимое выполним:

```
cat example.txt
```

Выведет все строки содержащие букву `a`

```
cat example.txt | grep a
```

Выведет все строки начинающиеся с буквы `a`

```
cat example.txt | grep ^a
```

Выведет все строки заканчивающиеся на `t`

```
cat example.txt | grep t$
```

Выведет строки где символ `p` идет 2 раза подряд

```
cat example.txt | grep -E p\{2}
```

Revision #6

Created 2021-09-22 15:07:07 UTC by gasick

Updated 2021-09-23 07:31:48 UTC by gasick