You may already know that in Unix, files have access settings and that there are several permissions and limitations for users. In this topic we will learn how to change access settings using commands in the Unix terminal. Let's find out how to do it below.

View file permissions

First of all, it's good to know what file permissions you already have. To check it, you can use the

`ls` command and its option `-l`. Just type

`ls -l <filename>`

in the terminal, and you will get something like this:

```
$ ls -l file.txt
```

```
-rw-r--r-- 1 admin admin 0 march 12 16:22 file.txt
```

Here you see who the file owner is, the creation date, and the permission set

`-rw-r--r--`

.

Also, you can use the `stat` command. It displays information about the files and file systems in more details than `ls -l`. The other difference is that file permissions here are displayed as octal numbers:

```
$ stat file.txt
  File: file.txt
  Size: 0           Blocks: 0          IO Block: 4096   regular
empty file
Device: 806h/2054d  Inode: 2114279     Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/   admin)   Gid: ( 1000/
admin)
```

```
Access: 2021-03-16 13:18:04.699986770 +0300
Modify: 2021-03-12 16:22:37.614043826 +0300
Change: 2021-05-19 19:50:50.112886819 +0300
```

```
 Birth: -
```

To shorten this output and view only access information you can use the

`grep` command. It finds and displays only those strings that contain the word *Access*:

```
$ stat file.txt | grep Access
Access: (0644/-rw-r--r--)  Uid: ( 1000/   admin)   Gid: ( 1000/
admin)
```

```
Access: 2021-03-16 13:18:04.699986770 +0300
```

In the same way, you can find out the permissions for the directory.

So, now you know how to see file permissions. Let's find out how you can change them.

Change file owner

If you want to change the owner of the file and/or a group, you can use the `chown` command. Its syntax is as follows:

```
chown user <option> /path/to/file
```

. In the user field, you need to specify the user to whom you want to delegate the file. You can also specify a group separated by a colon, for example, user: group. Then not only the user will change, but also the whole group. For example, let's take the

```
save_the_world.txt
```

and transfer it to *superman* user and also change the group:

```
chown superman:marvel ./save_the_world.txt
```

If you want to change the owner of an entire directory, you can use the same command, just write the path to the folder instead of the file path. The path can be specified as both absolute and relative as in the example above.

If you want these changes to apply not only to this directory but also to all its subdirectories, add the

`-R`

option:

```
chown -R superman:marvel ./save_the_world.txt
```

Great, now you know how to change the *owner* and the *group*. The next step is to learn how to "change the file mode" (another way to describe access permissions).

Change file mode

To change the file mode you can use the

`chmod`

command. With this command, you can set permissions to read, write, and execute a file/directory for the owner, group, and the world. The syntax of the command is as follows:

```
chmod permissions filename
```

There are two modes to use this command: an absolute mode and a symbolic mode. This means that we use either octal numbers or characters to define a permission set. Let's study some examples for the absolute mode first.

We have a file

`modify_it_now.exe` and its permissions are `-r--r--r--` or `444` in octal. The task is to change the mode so that the owner can read, write, and execute the file. The group should be able to read and execute, and the others should be able only to execute.

In the absolute mode it will look like this:

`751`

. So, you can type in the terminal:

```
chmod 751 modify_it_now.exe
```

Now let's move on to the symbolic mode.

Symbolic mode

As for the symbolic mode, you can modify permissions only of a specific owner. This mode involves mathematical symbols to modify the Unix file permissions:

- + adds permission to a file or a directory
- - removes permission
- = sets permission and overrides permissions set earlier

Also, it's important to know user denotations:

- u is user/owner
- g is group
- o is other
- a is all

The syntax thus will be as follows:

```
chmod <user denotation> <operator> <permissions>
```

.

So, if we want to modify the file permissions, we should type:

```
# modifying user permissions
chmod u=rwx modify_it_now.exe

# modifying group permissions
chmod g=r+x modify_it_now.exe

# modifying other permissions

chmod o=r+w modify_it_now.exe
```

Here we override the permissions so that users can read, write and execute, group can read and execute, and others can read and write.

In case we want to set the same permissions for all, for example, so that everybody could read, write and execute the file, we should type:

```
chmod a=rwx modify_it_now.exe
```

And if we want to only allow the others to read our file, we might use `+`:`chmod o+r modify_it_now.exe`

This way we do not override the whole permissions set but change the concrete permission. In the example above we added the reading permission to others without changing the rest.

Finally, if permission is to be removed, we should use the

`-`

operator.

Conclusion

To sum up,

- both `ls -l` or `stat` commands will give you the information about the file permissions;
- using `chown` command you can change the owner of the file and the group;
- with the `chmod` command you can change the mode of the file;
- remember the two ways to change the file mode: absolute and symbolic.

What command should we use to change the owner of all subdirectories in a directory? chown -R

What command should we use to change the file mode? Chmod

Jim has to change the owner and the group of a `test.txt` file. The new owner is admin, and the new group is team. Which variant of the command below will help him with this task?
chown admin:team ./test.txt

Nick wants to modify group permissions to a `program.exe` file so that the group members can read, write and execute it. Which variant of the command below should he use?
chmod g=rwx program.exe

Kate has a `work.py` file, and she needs to modify the permissions so that the user can read, write and execute the file and others can only read it. Kate wants to use the absolute mode to represent the permission set. Which set should she use? Use the table below to find out the right answer.

Number	Permission Type	Symbol
0	No Permission	---
1	Execute	--x
2	Write	-w-
3	Execute + Write	-wx
4	Read	r--
5	Read + Execute	r-x
6	Read +Write	rw-
7	Read + Write +Execute	rwx

744

What commands can we use to find out file permissions in Linux?
ls -l <filename>
stat <filename>

Imagine you have a project file called `/tmp/program.py`. Your task is to make this file executable for other users without overriding previous permissions. Use the `chmod` command and symbolic mode to do the task.

```
#!/usr/bin/env bash
solve() {
   chmod o+x /tmp/program.py

}
```