Copy files

First, let's take a look how to copy the file. Copy, means that we *copy* existing file to the new filename or location. The source file is still available.

In this lab we have some directories structure with files prepared. Let's take a look.

tree sourcedir

We used here a new command, tree. It recursively shows the content of the directory. Recursively means if there is a subdirectory, its content will be shown as well.

cp

The command to copy file is cp. The way how we use it is very simple:

cp source target

Ok, we know the command, let's use it. But first we need to create a directory where we will copy files.

mkdir targetdir

Let's go to our source directory

cd sourcedir

Now we can copy one file to the target directory. From our current location, the targeet is one level higher, so we need to use .. in order to say go to parent directory.

cp one ../targetdir

Ok, do we have the file there?

ls -l ../targetdir/one yes. Do we have the source file still located in the current directory?

ls -l one

Yeah! This works.

Copy and change the filename

Obviously, we can use cp not just for copying files, but also changing the name of the target file.

In the previous examply we copied one file. Let's copy this file again, but in order to have **two** files in the target directory.

cp one ../targetdir/another-one

Let's check

ls -l ../targetdir

Copy two or more files in one command

So, we know how to copy file. However, do we know how to copy two files, for example?

copy sourcefile1 targetfile1 and then copy sourcefile2 targetfile2? Well, yeah, we can do it this way. For three files too. For 10 files? Yeah, why not. For... ? :)

Ok, do not be hasty, we will come back to it. Right now let's take a look how we can copy two or so files in one shot.

This is what we will do:

cp file1tocopy file2copy file3tocopy targetlocation

So, last parameter describes where to copy all files listed in parameters (but last, ofc).

cp two01 two02 ../targetdir

Ok, do we have these files in targetdir?

ls -l ../targetdir

Yes, we have it.

Copy multiple files

We can copy many files in easy way, using wildcards.

In Linux we have multiple widlcards. We touch here only two.

* ? this wildcard replaces one character. So, if we wish to copy all files, where the file name start with my and ends with file, but there is something in between, like a, g or6, **but** it has to be only one character (so something like my3file, but not myFGfile), I can use ?. It will look like this my?file.
* \* this wildcard replaces all characters with nay length. Considering last example, if we wish to copy my1file and myFGfile, we will use my\*file.

If you wish to learn more about wildcards, run man 7 glob (and quit with q). We talk about man in another lab!

Ok, let's try it.

ls -l three\*

ls -l three\* | wc -l

We have *10* files, right?

cp three0? ../targetdir

ls -l ../targetdir/three\*

ls -l ../targetdir/three\* | wc -l

We have *9* files. Why? Please, try to find the answer yourself!

Copy directory structure

We copied files so far. Let's try to copy the whole directory structure, not just the file or files.

First we need to go to the parent directory.

cd

and now we can copy the whole structure:

cp -R sourcedir anotherdir

-R argument means *recursively*. How to understand this, please take look back on top when we discussed tree

ls -l anotherdir

We copied the whole structure.

Now, important thing. Please remember, the target directory must not exist. What will happen if the directory exist? Let's see!

mkdir testdir

Now we have the testdir directory, and we are ready to recursively copy all files from sourcedir. We do it in *exactly* the same way like previously.

cp -R sourcedir testdir

Recursive copy thought that you wish to copy *the whole structure****into****testdir directory*.

ls -l testdir

Here we see that the whole structure is copied under the directory.

ls -l testdir/\*

# Move files

Ok, we know ho to copy files. Sometimes however, we want to move the file from one place to another. For this case, we need to use mv command. The usage is quite similar to cp, but... simpler.

## Move one file

Let's clear the screen.

clear

Create another directory:

mkdir movedfiles

And finally we are ready to move our first file.

ls -l sourcedir/one  
mv sourcedir/one movedfiles  
ls -l sourcedir/one  
ls -l movedfiles

Done. Obviously, we can move and change the name:

ls -l anotherdir/one  
mv anotherdir/one movedfiles/another-one  
ls -l anotherdir/one  
ls -l movedfiles

## Move a few files

Similar like with cp we are able to move a few files to another location. It looks like this:

mv file1 file2 file3 targetlocation

Let's try:

mv sourcedir/two01 sourcedir/two02 movedfiles  
ls -l movedfiles

## Move directories

And finally, we can move all directories.

mv anotherdir newdir  
ls -l anotherdir  
ls -l newdir

And it works similarily like cp did when move is perform to existing directory:

mv newdir movedfiles  
ls -l newdir  
ls -l movedfiles  
ls -l movedfiles/newdir

BACKNEXT

# More possibilities

In order to copy file, or better to say, copy content of the file, we can use redirections.

We have a special file - .profile. Let's use this file to copy the content.

clear  
ls -l newfilewithcontent.txt

File is not exist.

cat .profile > newfilewithcontent.txt

ls -l newfilewithcontent.txt

Now we have the new file. Lets see, if the content was copied.

We can do it is simple way:

cat .profile  
cat newfilewithcontent.txt

In this way we visually confirmed that files are the same. But what if the file contains 10000 lines? Will we check every single line? Well, no.

We can check if there is any difference between two files. We do it with a new command - diff. This is the way to compare two files.

diff .profile newfilewithcontent.txt

No output means that both files are the same.

There are other methods to copy files. Even to copy and transform the content during the process. But this is the basic tutorial, so we are not touhing them.

ntu $ mkdir movedfiles

ubuntu $ ls -l sourcedir/one

ls: cannot access 'sourcedir/one': No such file or directory

ubuntu $ mv sourcedir/one movedfiles

mv: cannot stat 'sourcedir/one': No such file or directory

ubuntu $ ls -l sourcedir/one

ls: cannot access 'sourcedir/one': No such file or directory

ubuntu $ ls -l movedfiles

total 0

ubuntu $ ls -l anotherdir/one

ls: cannot access 'anotherdir/one': No such file or directory

ubuntu $ mv anotherdir/one movedfiles/another-one

mv: cannot stat 'anotherdir/one': No such file or directory

ubuntu $ ls -l anotherdir/one

ls: cannot access 'anotherdir/one': No such file or directory

ubuntu $ ls -l movedfiles

total 0

ubuntu $ mv sourcedir/two01 sourcedir/two02 movedfiles

mv: cannot stat 'sourcedir/two01': No such file or directory

mv: cannot stat 'sourcedir/two02': No such file or directory

ubuntu $ ls -l movedfiles

total 0

ubuntu $ mv anotherdir newdir

mv: cannot stat 'anotherdir': No such file or directory

ubuntu $ ls -l anotherdir

ls: cannot access 'anotherdir': No such file or directory

ubuntu $ ls -l newdir

ls: cannot access 'newdir': No such file or directory

ubuntu $ mv newdir movedfiles

mv: cannot stat 'newdir': No such file or directory

ubuntu $ ls -l newdir

ls: cannot access 'newdir': No such file or directory

ubuntu $ ls -l movedfiles

total 0

ubuntu $ ls -l movedfiles/newdir

ls: cannot access 'movedfiles/newdir': No such file or directory

ubuntu $ ls -l newfilewithcontent.txt

ls: cannot access 'newfilewithcontent.txt': No such file or directory

ubuntu $ cat .profile > newfilewithcontent.txt

cat: .profile: No such file or directory

ubuntu $ ls -l newfilewithcontent.txt

-rw-r--r-- 1 root root 0 Sep 5 11:40 newfilewithcontent.txt

ubuntu $ cat .profile

cat: .profile: No such file or directory

ubuntu $ cat newfilewithcontent.txt

ubuntu $ diff .profile newfilewithcontent.txt

diff: .profile: No such file or directory

ubuntu $ ^C

ubuntu $