Open Source Community

Session 2 Part 2

Agenda;

1. Man Pages
   1.1. What are Man pages?
   1.2. Searching for a Command
2. Managing Directories and Files
   2.1. Creating Directories
   2.2. File Extensions
   2.3. File manipulation
3. Links
   3.1. What are Links?
   3.2. Soft “Symbolic” Links
   3.3. Hard Links
   3.4. Deleting Links

1.Man page

1.1What is the man page

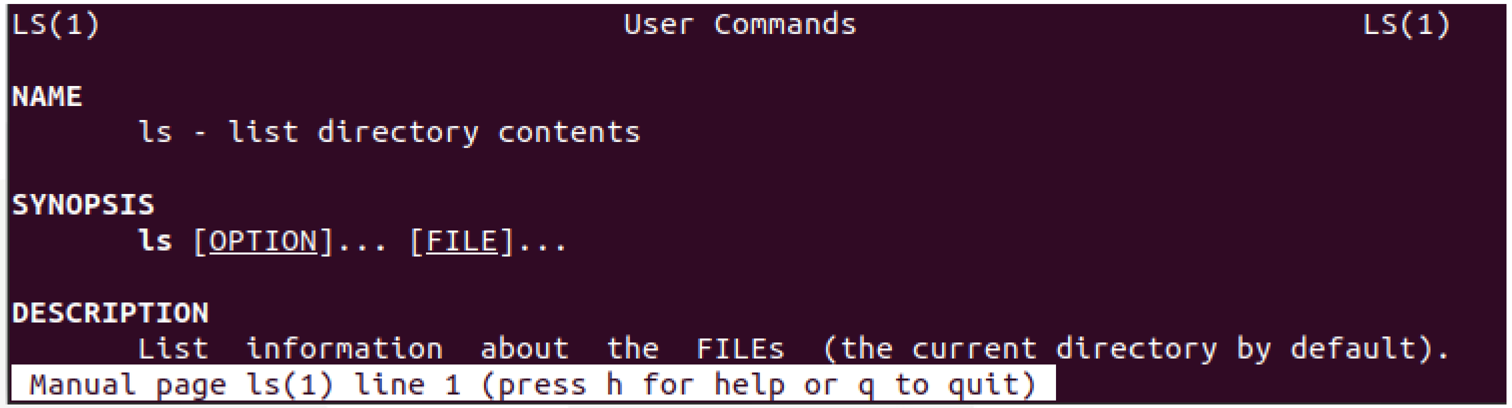
short for Manual Page

They’re a set of pages that explain what every command on the system does, what options are available, what arguments it can take, and shows you how to use them.

man [COMMAND NAME]

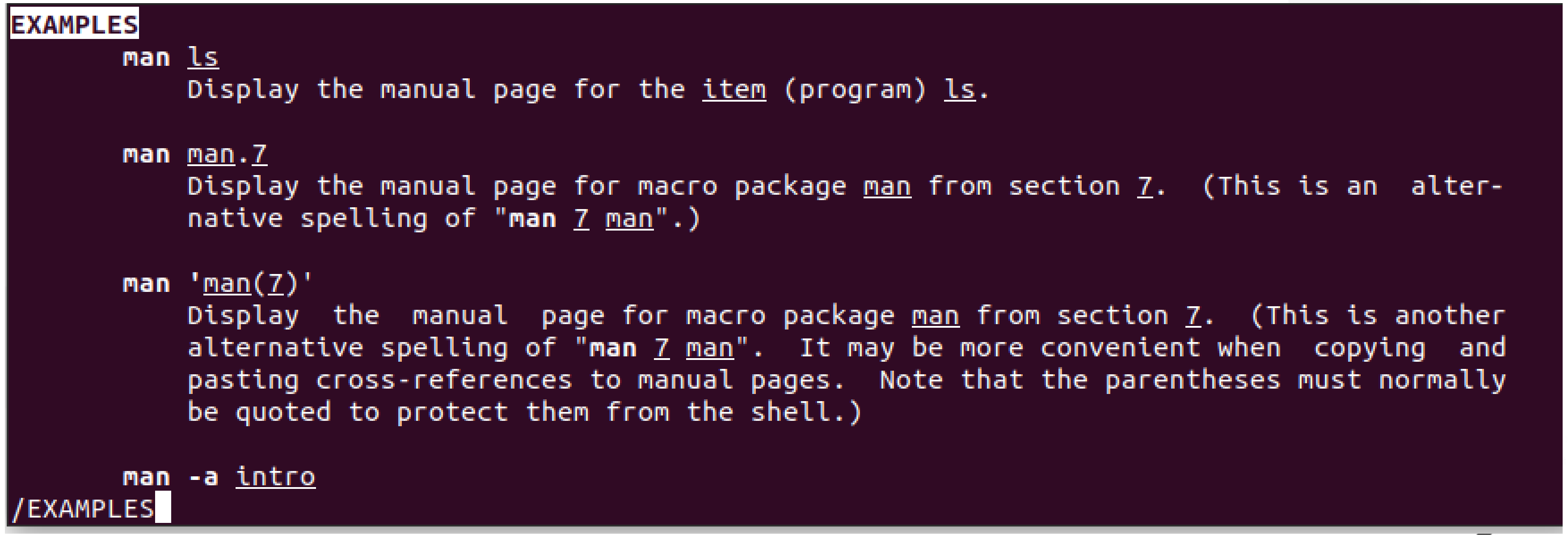
Ex: man ls This will show you the manual of the “ls” command.

You can use the arrow keys to navigate the pages, you can hit q to quit or h for help or / for search in the manual.



```
LS(1)                            User Commands                           LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about  the  FILEs  (the current directory by default).
 Manual page ls(1) line 1 (press h for help or q to quit)
```

Ex: man man   And search for “EXAMPLES"


```
EXAMPLES
       man ls
           Display the manual page for the item (program) ls.

       man man.7
           Display the manual page for macro package man from section 7.  (This is an  alter-
           native spelling of "man 7 man".)

       man 'man(7)'
           Display  the  manual  page for macro package man from section 7.  (This is another
           alternative spelling of "man 7 man".  It may be more convenient when  copying  and
           pasting cross-references to manual pages.  Note that the parentheses must normally
           be quoted to protect them from the shell.)

       man -a intro
/EXAMPLES
```

PS : You can also use info and --help to get information about a command

Ex: info ls , ls –help

1.2 searching for command

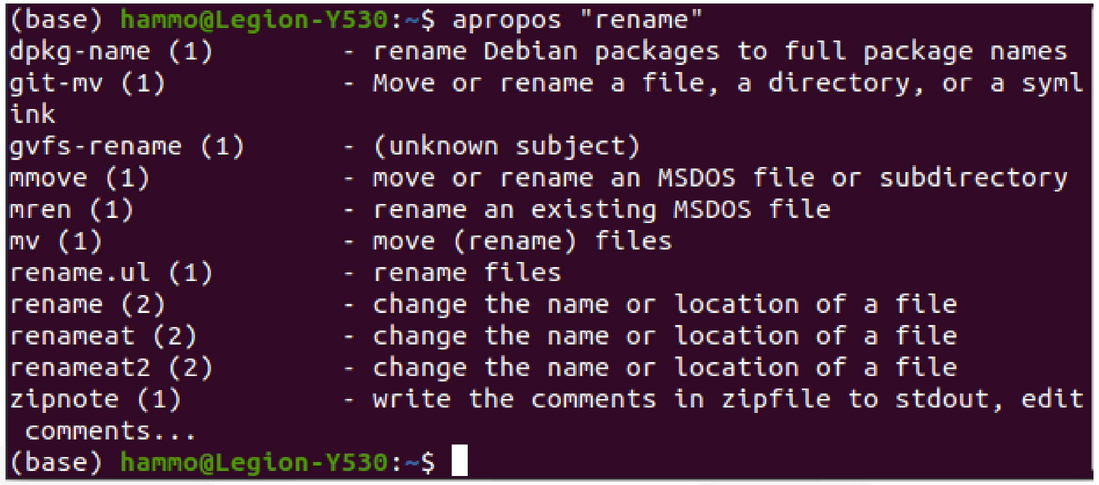
If you need to do a specific task but don’t know which command can do it.

apropos or man -k

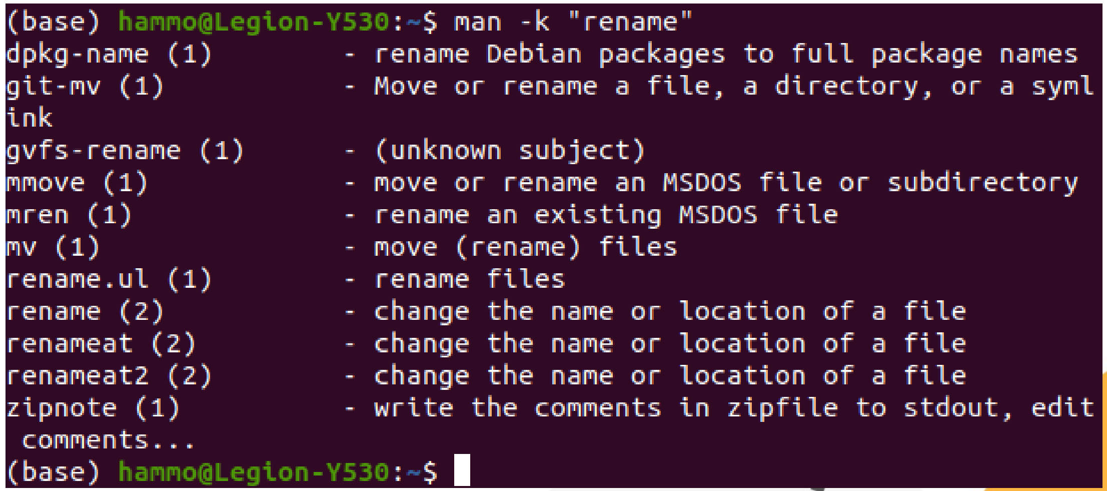
Are used fro searching for that command by the keyword of the command.

PS : You can use whatis or man -f to quickly see what a command does

Ex: You want to rename a file but you don’t know what command does that.



```
(base) hammo@Legion-Y530:~$ apropos "rename"
dpkg-name (1)        - rename Debian packages to full package names
git-mv (1)           - Move or rename a file, a directory, or a syml
ink
gvfs-rename (1)      - (unknown subject)
mmove (1)            - move or rename an MSDOS file or subdirectory
mren (1)             - rename an existing MSDOS file
mv (1)               - move (rename) files
rename.ul (1)        - rename files
rename (2)           - change the name or location of a file
renameat (2)         - change the name or location of a file
renameat2 (2)        - change the name or location of a file
zipnote (1)          - write the comments in zipfile to stdout, edit
 comments...
(base) hammo@Legion-Y530:~$
```



```
(base) hammo@Legion-Y530:~$ man -k "rename"
dpkg-name (1)        - rename Debian packages to full package names
git-mv (1)           - Move or rename a file, a directory, or a syml
ink
gvfs-rename (1)      - (unknown subject)
mmove (1)            - move or rename an MSDOS file or subdirectory
mren (1)             - rename an existing MSDOS file
mv (1)               - move (rename) files
rename.ul (1)        - rename files
rename (2)           - change the name or location of a file
renameat (2)         - change the name or location of a file
renameat2 (2)        - change the name or location of a file
zipnote (1)          - write the comments in zipfile to stdout, edit
 comments...
(base) hammo@Legion-Y530:~$
```

As you can see, all of these are commands that can rename something. Which one you choose is up to you, you can use man pages to know more about these commands and select the suitable one for your case.

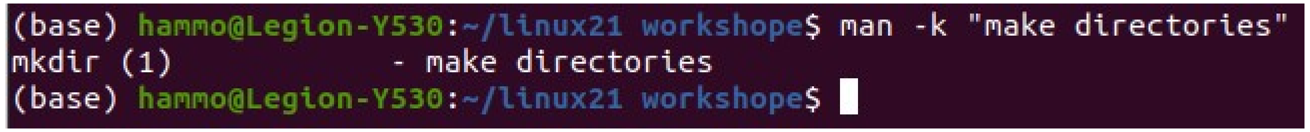
2.Managing Directories and Files

2.1 .Creating Directories

You now know how to use man. Can you search for a command that creates directories?

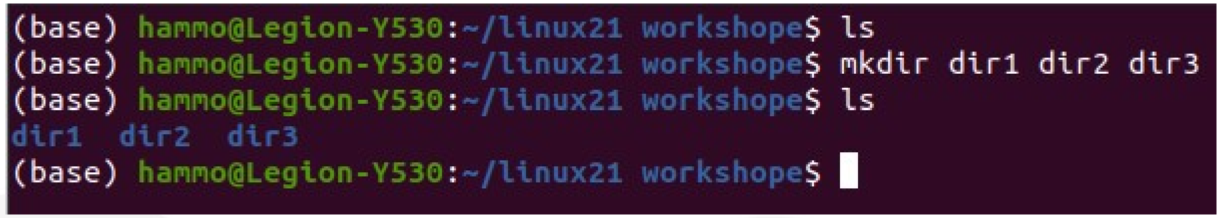
Note: We use double quotes " " if the name of the directory has more than one word. This is to avoid making the shell interpret the 2 words as 2 separate arguments

Solution : man -k "make directories"   and the command is mkdir



```
(base) hammo@Legion-Y530:~/linux21 workshope$ man -k "make directories"
mkdir (1)            - make directories
(base) hammo@Legion-Y530:~/linux21 workshope$ 
```

You can pass single argument or many



```
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
(base) hammo@Legion-Y530:~/linux21 workshope$ mkdir dir1 dir2 dir3
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
dir1  dir2  dir3
(base) hammo@Legion-Y530:~/linux21 workshope$ 
```

2.2File Extensions

A file extension is the ending of a file’s name that helps the operating system and the user know the kind of file it is and what program should run to open this file.

By default Linux has 3 types of files:

1.Regular File (-):

- Readable file (.txt, .cpp)
- Binary file (.exe)
- Image file (.png, .jpg)
- Archive or “Compressed” file (.zip, .rar, .doc, .pdf)

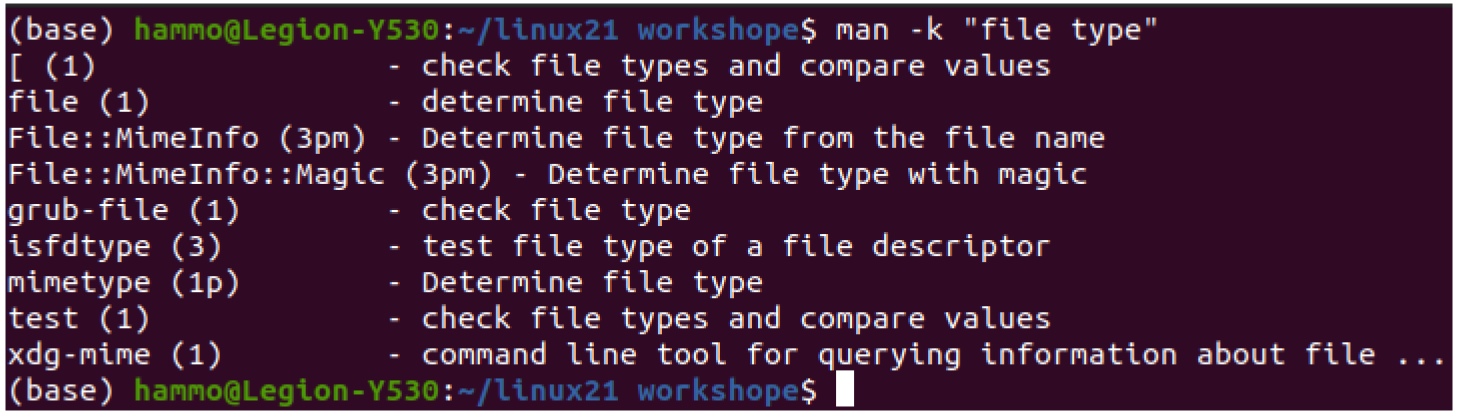
2. Directory (d): A folder containing other files or folders

3. Special

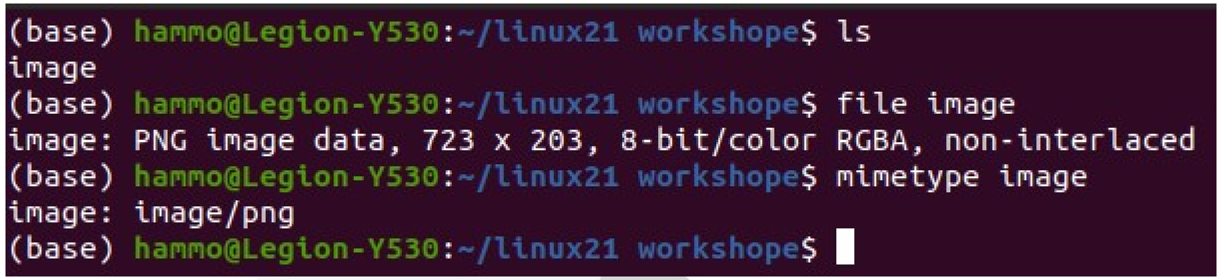
- Block File (b): Hardware files (Like some files under /dev/)
- Soft “Symbolic” Link File (l): File pointing to another file (shortcut)

Using the man page how to determine
the file type?

Solution : man -k "file type"   and the commands are mimetype or file



```
(base) hammo@Legion-Y530:~/linux21 workshope$ man -k "file type"
[ (1)                - check file types and compare values
file (1)             - determine file type
File::MimeInfo (3pm) - Determine file type from the file name
File::MimeInfo::Magic (3pm) - Determine file type with magic
grub-file (1)        - check file type
isfdtype (3)         - test file type of a file descriptor
mimetype (1p)        - Determine file type
test (1)             - check file types and compare values
xdg-mime (1)         - command line tool for querying information about file ...
(base) hammo@Legion-Y530:~/linux21 workshope$
```

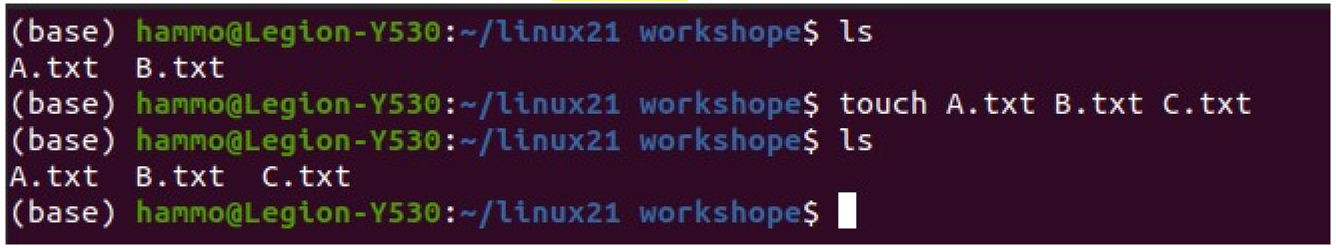


```
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
image
(base) hammo@Legion-Y530:~/linux21 workshope$ file image
image: PNG image data, 723 x 203, 8-bit/color RGBA, non-interlaced
(base) hammo@Legion-Y530:~/linux21 workshope$ mimetype image
image: image/png
(base) hammo@Legion-Y530:~/linux21 workshope$
```

2.3. File manipulation

CREATING FILES

To create file use touch command

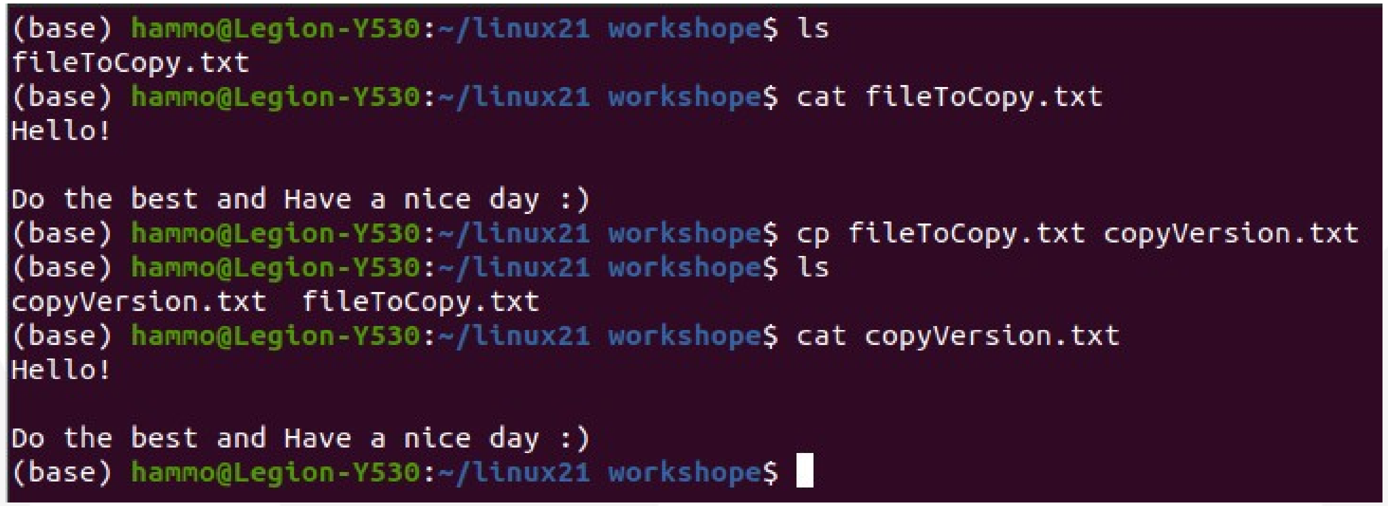


```
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
A.txt  B.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ touch A.txt B.txt C.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
A.txt  B.txt  C.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ 
```

Note : Like mkdir you can pass single argument or many

COPYING FILES

To copy file use cp command   cp <original file> <copy file>

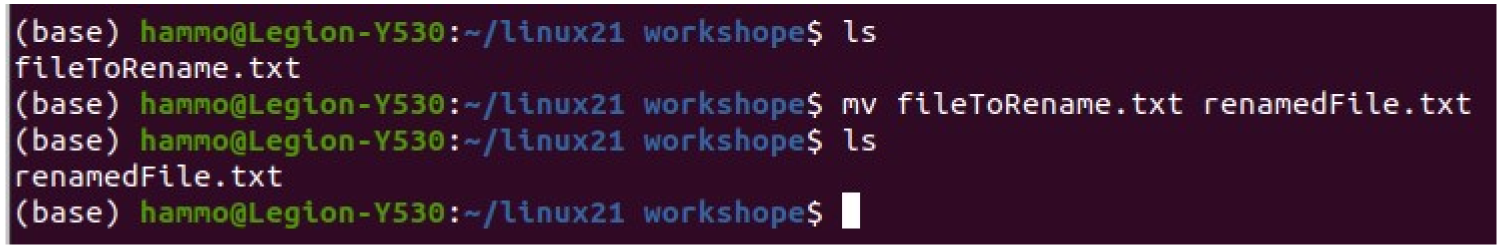


```
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
fileToCopy.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ cat fileToCopy.txt
Hello!

Do the best and Have a nice day :)
(base) hammo@Legion-Y530:~/linux21 workshope$ cp fileToCopy.txt copyVersion.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
copyVersion.txt  fileToCopy.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ cat copyVersion.txt
Hello!

Do the best and Have a nice day :)
(base) hammo@Legion-Y530:~/linux21 workshope$ 
```

Renaming and Moving Files

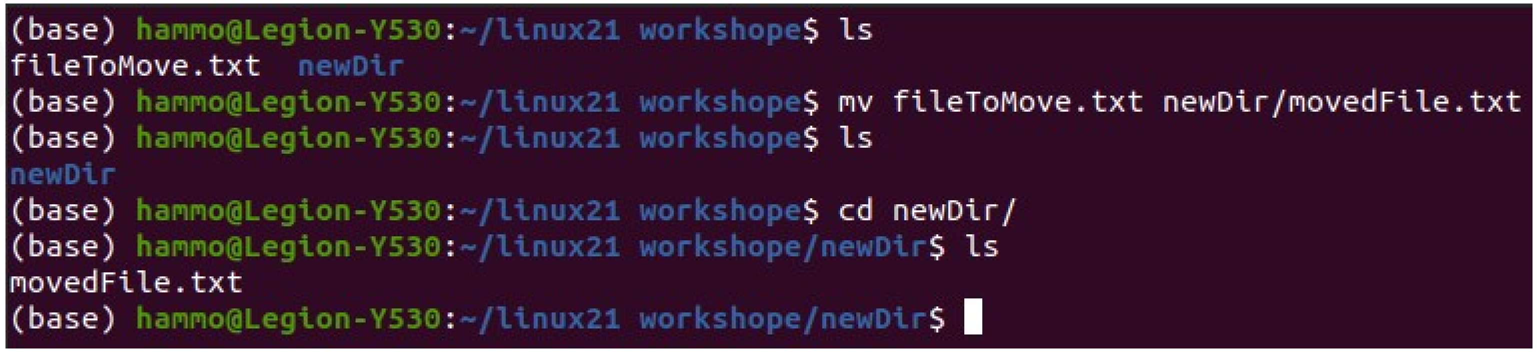
To rename a file use mv

mv <original file> <copy file>



```
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
fileToRename.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ mv fileToRename.txt renamedFile.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
renamedFile.txt
(base) hammo@Legion-Y530:~/linux21 workshope$
```

To move “cut” a file use also mv



```
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
fileToMove.txt  newDir
(base) hammo@Legion-Y530:~/linux21 workshope$ mv fileToMove.txt newDir/movedFile.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
newDir
(base) hammo@Legion-Y530:~/linux21 workshope$ cd newDir/
(base) hammo@Legion-Y530:~/linux21 workshope/newDir$ ls
movedFile.txt
(base) hammo@Legion-Y530:~/linux21 workshope/newDir$
```

Deleting Files and Directories

You can delete a file or a directory using the rm command.

Try to figure it out from the man pages on your own before see the solution.

- Delete a file.
- Delete an empty directory.
- Delete a non empty directory.
- Delete but don’t act until the user confirms.
- Delete by force and don’t prompt the user

solution

- rm <file name>
- rm -d <dir name> or rmdir <dir name>
- rm -r <dir name>
- rm -i <file name>
- rm -f <file name>

3.Links

3.1. What are Links?

A link in Linux is a file that points to another file/directory. Creating links is similar to creating shortcuts. A file can have multiple links linked to it. But a link can only be linked to (pointed to) one file.

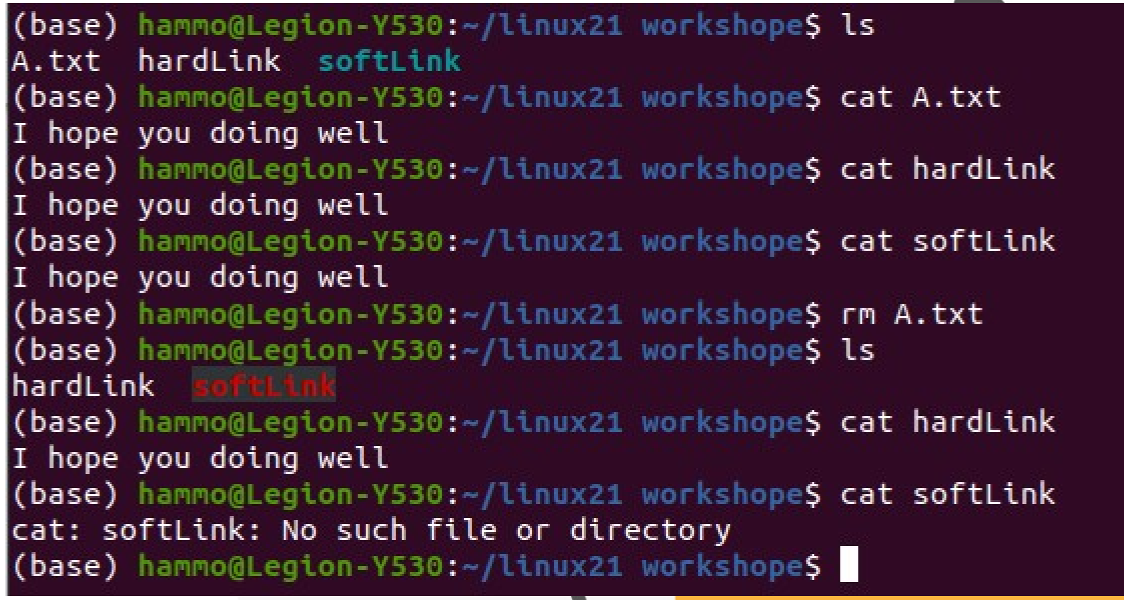
Note: You can think of links like pointers in Programming languages, if you're familiar with them.

There are two types of links:
(1)Soft or Symbolic links.
(2)Hard links.

These links behave differently when the source of the link (what is being linked to) is moved or removed.

- Symbolic links are not updated, and become "hanging links".
- Hard links always refer to the source, even if moved or removed.

Ex : We have a file A.txt. If we create a soft link and a hard link both pointing to it and then delete A.txt, the result is visible in the opposite figure.



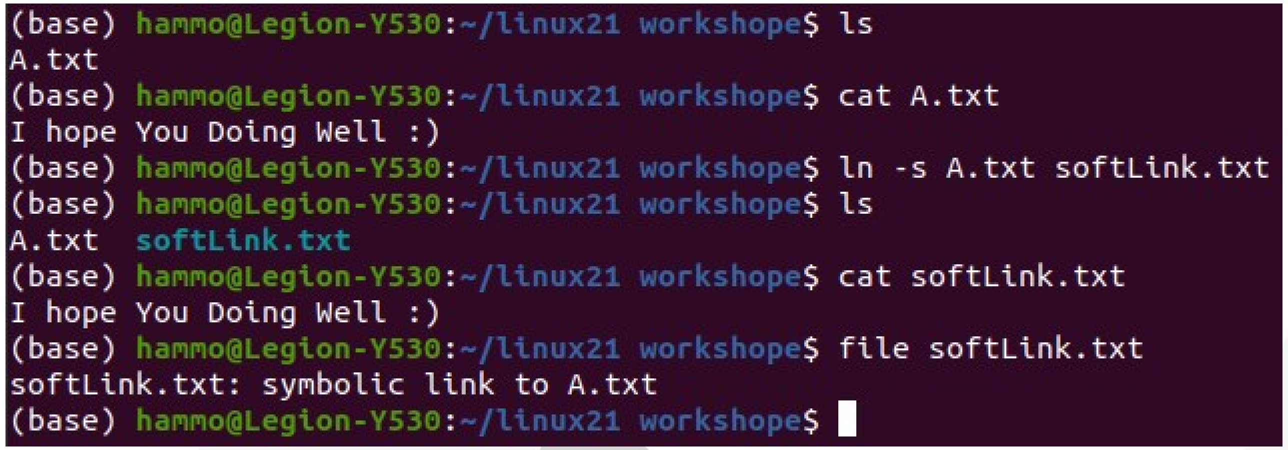
```
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
A.txt  hardLink  softLink
(base) hammo@Legion-Y530:~/linux21 workshope$ cat A.txt
I hope you doing well
(base) hammo@Legion-Y530:~/linux21 workshope$ cat hardLink
I hope you doing well
(base) hammo@Legion-Y530:~/linux21 workshope$ cat softLink
I hope you doing well
(base) hammo@Legion-Y530:~/linux21 workshope$ rm A.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
hardLink  softLink
(base) hammo@Legion-Y530:~/linux21 workshope$ cat hardLink
I hope you doing well
(base) hammo@Legion-Y530:~/linux21 workshope$ cat softLink
cat: softLink: No such file or directory
(base) hammo@Legion-Y530:~/linux21 workshope$
```

We can simply say that a soft link is just a file that points to another (shortcut), while a hard link is a copy of a file that is always synchronized with it.

3.2. Soft “symbolic” link

- A soft link is similar to the file shortcut feature which is used in Windows operating systems.
- Soft links can be linked across different filesystems, although if the original file is deleted or moved, the soft link will not work correctly and is referred to as a “hanging link”.
- Soft links contain the path for original file but not the content.
- A soft link can link to a directory.

ln -s <file name> <link name>



```
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
A.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ cat A.txt
I hope You Doing Well :)
(base) hammo@Legion-Y530:~/linux21 workshope$ ln -s A.txt softLink.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
A.txt  softLink.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ cat softLink.txt
I hope You Doing Well :)
(base) hammo@Legion-Y530:~/linux21 workshope$ file softLink.txt
softLink.txt: symbolic link to A.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ 
```

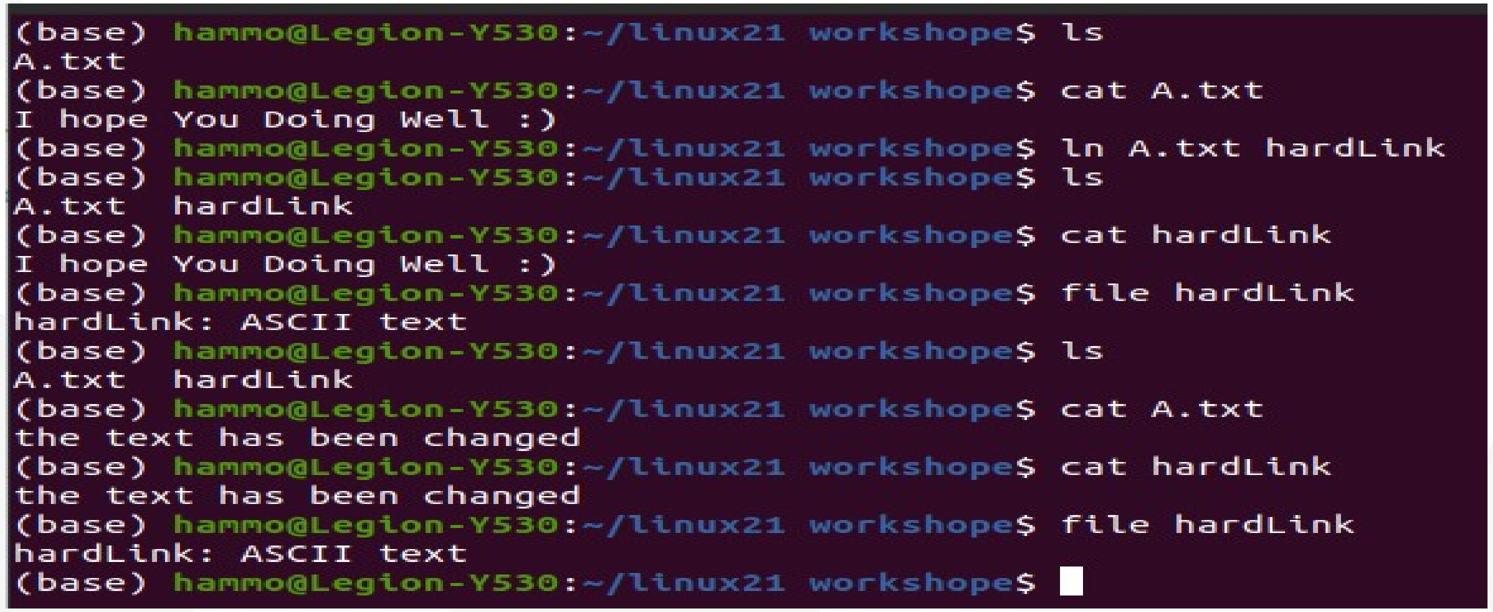
3.3.hard link

- A hard link is similar to creating a copy that is always synced with the original file.
- Hard links can’t be made across different file systems. But, if the original file is deleted or moved, the hard link will still work.
- A hard link can’t link to a directory.

In this example you can see that hardLink is treated as a normal file, it is just linked to A.txt. After editing A.txt, hardLink was edited too.

It is exactly like a synced copy of the file.

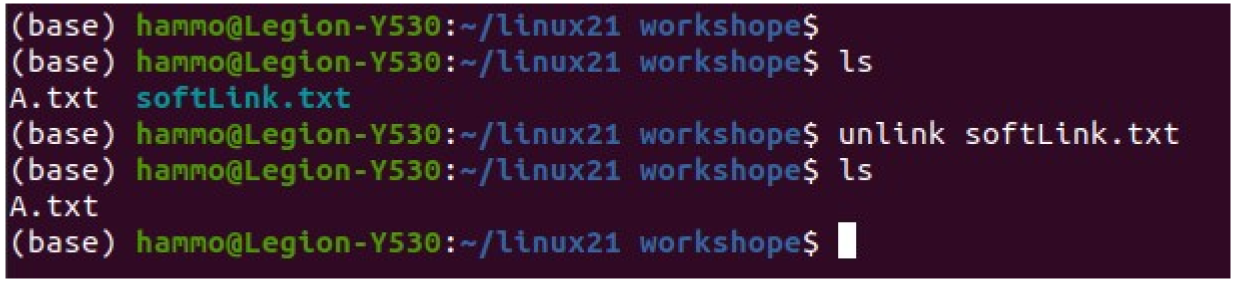
ln <file name> <link name>



```
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
A.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ cat A.txt
I hope You Doing Well :)
(base) hammo@Legion-Y530:~/linux21 workshope$ ln A.txt hardLink
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
A.txt  hardLink
(base) hammo@Legion-Y530:~/linux21 workshope$ cat hardLink
I hope You Doing Well :)
(base) hammo@Legion-Y530:~/linux21 workshope$ file hardLink
hardLink: ASCII text
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
A.txt  hardLink
(base) hammo@Legion-Y530:~/linux21 workshope$ cat A.txt
the text has been changed
(base) hammo@Legion-Y530:~/linux21 workshope$ cat hardLink
the text has been changed
(base) hammo@Legion-Y530:~/linux21 workshope$ file hardLink
hardLink: ASCII text
(base) hammo@Legion-Y530:~/linux21 workshope$ 
```

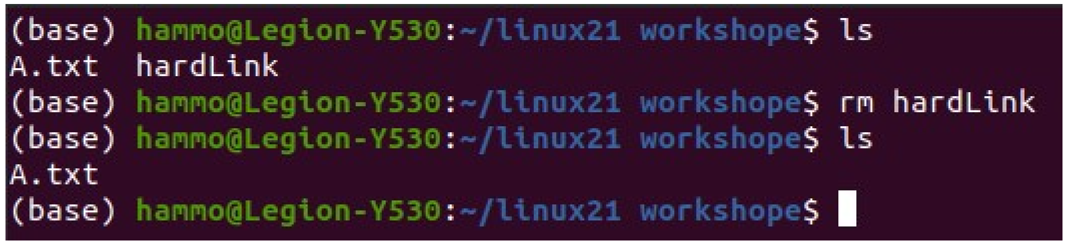
3.4. delete link

unlink



```
(base) hammo@Legion-Y530:~/linux21 workshope$
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
A.txt  softLink.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ unlink softLink.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
A.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ 
```

And since a link is a file after all, you can also delete it with rm.



```
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
A.txt  hardLink
(base) hammo@Legion-Y530:~/linux21 workshope$ rm hardLink
(base) hammo@Legion-Y530:~/linux21 workshope$ ls
A.txt
(base) hammo@Legion-Y530:~/linux21 workshope$ 
```

Thank you

#Stay_Safe