

RHSA (Admin 1) Red Hat Enterprise Linux 8 Essentials

Prepared by Eng. Taghreed Elsayed

Lecture 2

جامعة الجميع الذكية

تسعى لتكون الكيان الأول الذي يتبنى مبدأ المجتمع التعليمي
التعاوني ويقدم العلم مقابل العمل لتنمية وبناء المجتمعات.

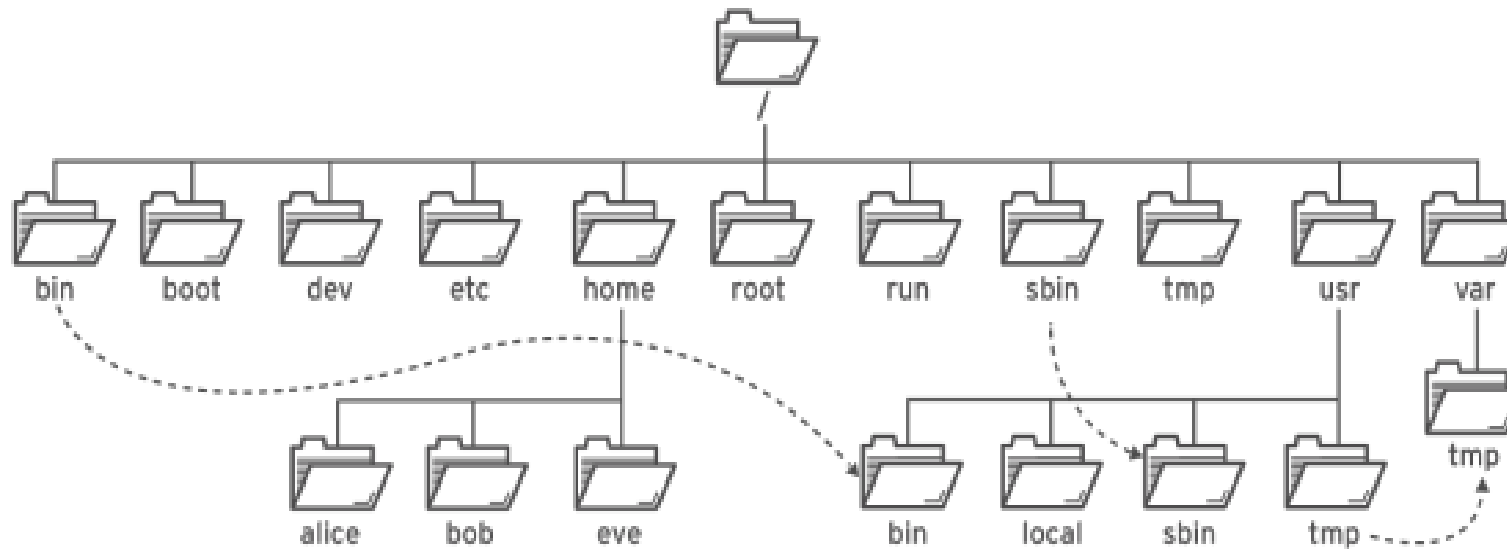
Managing Files from the Command Line



جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY

THE FILE SYSTEM HIERARCHY:

- ❑ All files on a Linux system are stored on file systems which are organized into a single inverted tree of directories, known as a file system hierarchy.
- ❑ This tree is inverted because the root of the tree is said to be at the top of the hierarchy, and the branches of directories and subdirectories stretch below the root.



Significant file system directories in Red Hat

- ❑ The directory / is the root directory at the top of the file system hierarchy.
- ❑ The / character is also used as a directory separator in file names. For example, if etc is a subdirectory of the / directory, we could call that directory /etc.
- ❑ Likewise, if the / etc directory contained a file named issue, we could refer to that file as / etc/issue.
- ❑ Subdirectories of / are used for standardized purposes to organize files by type and purpose. This makes it easier to find files. For example, in the root directory, the subdirectory /boot is used for storing files needed to boot the system.

Managing Files from the Command Line

Important Red Hat Enterprise Linux directories

LOCATION	PURPOSE
/usr	Installed software, shared libraries, include files, and static read-only program data. Important subdirectories include: - /usr/bin: User commands. - /usr/sbin: System administration commands. - /usr/local: Locally customized software.
/etc	Configuration files specific to this system.
/var	Variable data specific to this system that should persist between boots. Files that dynamically change (e.g. databases, cache directories, log files, printer-spoiled documents, and website content) may be found under /var .
/run	Runtime data for processes started since the last boot. This includes process ID files and lock files, among other things. The contents of this directory are recreated on reboot. (This directory consolidates /var/run and /var/lock from older versions of Red Hat Enterprise Linux.)
/home	Home directories where regular users store their personal data and configuration files.
/root	Home directory for the administrative superuser, root.
/tmp	A world-writable space for temporary files. Files which have not been accessed, changed, or modified for 10 days are deleted from this directory automatically. Another temporary directory exists, /var/tmp , in which files that have not been accessed, changed, or modified in more than 30 days are deleted automatically.
/boot	Files needed in order to start the boot process.
/dev	Contains special <i>device files</i> which are used by the system to access hardware.

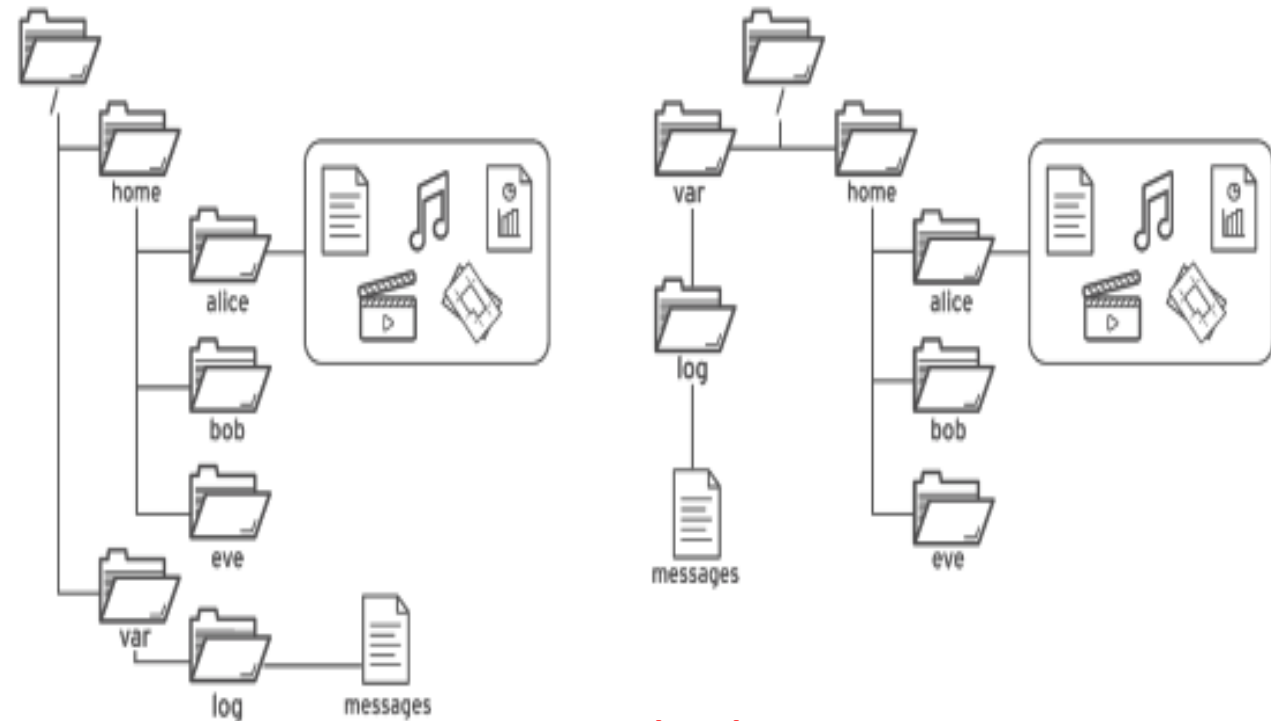


FILE SYSTEM HIERARCHY

DIRECTORY PURPOSE	LOCATION
This directory contains static, persistent system configuration data.	/etc
This is the system's root directory.	/
User home directories are located under this directory.	/home
This is the root account's home directory.	/root
This directory contains dynamic configuration data, such as FTP and websites.	/var
Regular user commands and utilities are located here.	/usr/bin
System administration binaries, for root use, are here.	/usr/sbin
Temporary files are stored here.	/tmp
Contains dynamic, non-persistent application runtime data.	/run
Contains installed software programs and libraries.	/usr

❖ ABSOLUTE PATHS AND RELATIVE PATHS

- The path of a file or directory specifies its unique file system location.
- Following a file path traverses one or more named subdirectories, delimited by a forward slash (/), until the destination is reached.
- Standard file behavior definitions apply to directories (also called folders) the same as other file types.



The common file browser view (left) is equivalent to the top-down view (right)

Absolute paths : An absolute path is a fully qualified name, beginning at the root (/) directory and specifying each subdirectory traversed to reach and uniquely represent a single file.

- ❑ Every file in a file system has a unique absolute path name, recognized with a simple rule: A path name with a forward slash (/) as the first character is an absolute path name.
- ❑ For example, the absolute path name for the system message log file is /var/log/messages. Absolute path names can be long to type, so files may also be located relatively.
- ❑ When a user logs in and opens a command window, the initial location is normally the user's home directory. System processes also have an initial directory.
- ❑ Users and processes navigate to other directories as needed; the terms working directory or current working directory refer to their current location.

- ❑ **Relative paths** Like an absolute path, a relative path identifies a unique file, specifying only the path necessary to reach the file from the working directory.
- ❑ Recognizing relative path names follows a simple rule: A path name with anything other than a forward slash (/) as a first character is a relative path name.
- ❑ A user in the /var directory could refer to the message log file relatively as log/messages. For standard Linux file systems, the path name of a file, including all / characters, may be no more than 4095 bytes long.
- ❑ Each component of the path name separated by / characters may be no more than 255 bytes long. File names can use any UTF-8 encoded Unicode character except / and the NUL character.

Managing Files from the Command Line

- ❑ (ASCII characters require one byte; other Latin, Greek, Hebrew, or Cyrillic characters take two bytes; remaining characters in the Unicode Basic Multilingual Plane take three; and no character will take more than four bytes.) Linux file systems—including, but not limited to, **ext4, XFS, BTRFS, GFS2, and Gluster FS**—are case-sensitive.
- ❑ Creating FileCase.txt and filecase.txt in the same directory results in two unique files. Although many non-Linux file systems are supported in Linux, each has unique file naming rules. For example, the ubiquitous VFAT file system is not case-sensitive and allows only one of the two example files to be created.
- ❑ However, VFAT, along with Microsoft's NTFS and Apple's HFS+, has case preserving behavior. Although these file systems are not case-sensitive (enforced primarily to support backward compatibility), they do display file names with the original capitalization used when the file was created.

NAVIGATING PATHS The `pwd` command displays the full path name of the current location, which helps determine appropriate syntax for reaching files using relative path names.

- The `ls` command lists directory contents for the specified directory or, if no directory is given, for the current directory.

```
[student@desktopX ~]$ pwd
/home/student
[student@desktopX ~]$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
[student@desktopX ~]$
```

- Use the `cd` command to change directories.
- With a working directory of `/home/student`, relative path syntax is shortest to reach the `Videos` subdirectory. The `Documents` subdirectory is then reached using absolute path syntax.

Managing Files from the Command Line

```
[student@desktopX ~]$ cd Videos [student@desktopX Videos]$ pwd
/home/student/Videos
[student@desktopX Videos]$ cd /home/student/Documents
[student@desktopX Documents]$ pwd
/home/student/Documents
[student@desktopX Documents]$ cd
[student@desktopX ~]$ pwd
/home/student
[student@desktopX ~]$
```

- ❑ **The touch command normally updates a file's timestamp to the** current date and time without otherwise modifying it. This is useful for creating empty files, which can be used for practice, since "touching" a file name that does not exist causes the file to be created. Using touch, practice files are created in the Documents and Videos subdirectories.

```
[student@desktopX ~]$
touch Videos/blockbuster1.ogg
[student@desktopX ~]$
touch Videos/blockbuster2.ogg
[student@desktopX ~]$
touch Documents/thesis_chapter1.odf
[student@desktopX ~]$
touch Documents/thesis_chapter2.odf
[student@desktopX ~]$
```

- ❖ The **ls** command has multiple options for displaying attributes on files.
- ❖ The most common and useful are **-l** (long listing format), **-a** (all files, includes hidden files), and **-R** (recursive, to include the contents of all subdirectories).

```
[student@desktopX ~]$ ls -l
total 15
drwxr-xr-x. 2 student student 4096 Feb  7 14:02 Desktop
drwxr-xr-x. 2 student student 4096 Jan  9 15:00 Documents
drwxr-xr-x. 3 student student 4096 Jan  9 15:00 Downloads
drwxr-xr-x. 2 student student 4096 Jan  9 15:00 Music
drwxr-xr-x. 2 student student 4096 Jan  9 15:00 Pictures
drwxr-xr-x. 2 student student 4096 Jan  9 15:00 Public
drwxr-xr-x. 2 student student 4096 Jan  9 15:00 Templates
drwxr-xr-x. 2 student student 4096 Jan  9 15:00 Videos
[student@desktopX ~]$ ls -la
total 15
drwx-----. 16 student student 4096 Feb  8 16:15 .
drwxr-xr-x.  6 root    root    4096 Feb  8 16:13 ..
-rw-----.  1 student student 22664 Feb  8 00:37 .bash_history
-rw-r--r--.  1 student student   18 Jul  9  2013 .bash_logout
-rw-r--r--.  1 student student  176 Jul  9  2013 .bash_profile
-rw-r--r--.  1 student student  124 Jul  9  2013 .bashrc
drwxr-xr-x.  4 student student 4096 Jan 20 14:02 .cache
drwxr-xr-x.  8 student student 4096 Feb  5 11:45 .config
drwxr-xr-x.  2 student student 4096 Feb  7 14:02 Desktop
drwxr-xr-x.  2 student student 4096 Jan  9 15:00 Documents
drwxr-xr-x.  3 student student 4096 Jan 25 20:48 Downloads
drwxr-xr-x. 11 student student 4096 Feb  6 13:07 .gnome2
drwx-----.  2 student student 4096 Jan 20 14:02 .gnome2_private
-rw-----.  1 student student 15190 Feb  8 09:49 .ICEauthority
drwxr-xr-x.  3 student student 4096 Jan  9 15:00 .local
drwxr-xr-x.  2 student student 4096 Jan  9 15:00 Music
drwxr-xr-x.  2 student student 4096 Jan  9 15:00 Pictures
drwxr-xr-x.  2 student student 4096 Jan  9 15:00 Public
drwxr-xr-x.  2 student student 4096 Jan  9 15:00 Templates
drwxr-xr-x.  2 student student 4096 Jan  9 15:00 Videos
[student@desktopX ~]$
```

Managing Files from the Command Line

```
[student@desktopX ~]$ls -R
.:
Desktop Documents Downloads Music Pictures Public Templates Videos

./Desktop:

./Documents:
thesis_chapter1.odf thesis_chapter2.odf

./Downloads:

./Music:

./Pictures:

./Public:

./Templates:

./Videos:
blockbuster1.ogg blockbuster2.ogg
[student@desktopX ~]$
```



The **cd** command has many options.

- ❑ A few are so useful as to be worth practicing early and using often. The command `cd -` changes directory to the directory where the user was previous to the current directory. Watch as this user takes advantage of this behavior to alternate between two directories, useful when processing a series of similar tasks.

```
[student@desktopX ~]$ cd Videos [student@desktopX Videos]$ pwd
/home/student/Videos
[student@desktopX Videos]$ cd /home/student/Documents
[student@desktopX Documents]$ pwd
/home/student/Documents
[student@desktopX Documents]$ cd -
[student@desktopX Videos]$ pwd
/home/student/Videos
[student@desktopX Videos]$ cd -
[student@desktopX Documents]$ pwd
/home/student/Documents
[student@desktopX Documents]$ cd -
[student@desktopX Videos]$ pwd
/home/student/Videos
[student@desktopX Videos]$ cd
[student@desktopX ~]$
```

LOCATING FILES AND DIRECTORIES

ACTION TO ACCOMPLISH	COMMAND
List the current user's home directory (long format) in simplest syntax, when it is not the current location.	<code>ls -l ~</code>
Return to the current user's home directory.	<code>cd</code>
Determine the absolute path name of the current location.	<code>pwd</code>
Return to the most previous working directory.	<code>cd -</code>
Move up two levels from the current location.	<code>cd ../../</code>
List the current location (long format) with hidden files.	<code>ls -al</code>
Move to the binaries location, from any current location.	<code>cd /bin</code>
Move up to the parent of the current location.	<code>cd ..</code>
Move to the binaries location, from the root directory.	<code>cd bin</code>

File management commands

ACTIVITY	SINGLE SOURCE ^(NOTE)	MULTIPLE SOURCE ^(NOTE)
Copy file	cp file1 file2	cp file1 file2 file3 dir ⁽⁴⁾
Move file	mv file1 file2 ⁽¹⁾	mv file1 file2 file3 dir ⁽⁴⁾
Remove file	rm file1	rm -f file1 file2 file3 ⁽⁵⁾
Create directory	mkdir dir	mkdir -p par1/par2/dir ⁽⁶⁾
Copy directory	cp -r dir1 dir2 ⁽²⁾	cp -r dir1 dir2 dir3 dir4 ⁽⁴⁾
Move directory	mv dir1 dir2 ⁽³⁾	mv dir1 dir2 dir3 dir4 ⁽⁴⁾
Remove directory	rm -r dir1 ⁽²⁾	rm -rf dir1 dir2 dir3 ⁽⁵⁾
Remove empty directory	rmdir dir1	rmdir -p dir1/dir2/dir3

Create directories The **mkdir** command

- ❑ creates one or more directories or subdirectories, generating errors if the file name already exists or when attempting to create a directory in a parent directory that doesn't exist.
- ❑ The **-p** parent option creates missing parent directories for the requested destination. Be cautious when using **mkdir -p**, since accidental spelling mistakes create unintended directories without generating error messages.
- ❑ In the following example, a user attempts to use **mkdir** to create a subdirectory named **Watched** in the existing **Videos** directory, but mistypes the directory name.

```
[student@desktopX ~]$mkdir Video/Watched
mkdir: cannot create directory `Video/Watched': No such file or directory
```

The **mkdir** failed because **Videos** was misspelled and the directory **Video** does not exist. If the user had used **mkdir** with the **-p** option, there would be no error and the user would end up with two directories, **Videos** and **Video**, and the **Watched** subdirectory would be created in the wrong place.

Chapter 03

Managing Files from the Command Line

```
[student@desktopX ~]$mkdir Videos/Watched[student@desktopX ~]$cd
Documents[student@desktopX Documents]$mkdir ProjectX ProjectY[student@desktopX
Documents]$mkdir -p Thesis/Chapter1 Thesis/Chapter2 Thesis/
Chapter3[student@desktopX Documents]$cd[student@desktopX ~]$ls -R Videos Documents
Documents:
ProjectX ProjectY Thesis thesis_chapter1.odf thesis_chapter2.odf

Documents/ProjectX:

Documents/ProjectY:

Documents/Thesis:
Chapter1 Chapter2 Chapter3

Documents/Thesis/Chapter1:

Documents/Thesis/Chapter2:

Documents/Thesis/Chapter3:

Videos:
blockbuster1.ogg blockbuster2.ogg Watched

Videos/Watched:

[student@desktopX ~]$
```

Copy files The **cp** command **copies one** or more files to become new, independent files. Syntax allows copying an existing file to a new file in the current or another directory, or copying multiple files into another directory.

- ❑ In any destination, new file names must be unique. If the new file name is not unique, the copy command will overwrite the existing file.

```
[student@desktopX ~]$cd Videos[student@desktopX Videos]$cp blockbuster1.ogg  
blockbuster3.ogg[student@desktopX Videos]$ls -l  
total 0  
-rw-rw-r--. 1 student student  0 Feb  8 16:23 blockbuster1.ogg  
-rw-rw-r--. 1 student student  0 Feb  8 16:24 blockbuster2.ogg  
-rw-rw-r--. 1 student student  0 Feb  8 19:02 blockbuster3.ogg  
drwxrwxr-x. 2 student student 4096 Feb  8 23:35 Watched  
[student@desktopX Videos]$
```

- ❖ To protect users from accidentally overwriting directories with contents, multiple file cp commands ignore directories specified as a source.
- ❖ Copying non-empty directories, with contents, requires the -r recursive option.

```
[student@desktopX Videos]$cd ../Documents[student@desktopX Documents]$cp
thesis_chapter1.odf thesis_chapter2.odf Thesis ProjectX
cp: omitting directory `Thesis'
[student@desktopX Documents]$ cp -r Thesis ProjectX
[student@desktopX Documents]$ cp thesis_chapter2.odf Thesis/Chapter2/
[student@desktopX Documents]$ ls -R
.:
ProjectX ProjectY Thesis thesis_chapter1.odf thesis_chapter2.odf

./ProjectX:
Thesis thesis_chapter1.odf thesis_chapter2.odf

./ProjectX/Thesis:

./ProjectY:

./Thesis:
Chapter1 Chapter2 Chapter3

./Thesis/Chapter1:

./Thesis/Chapter2:
thesis_chapter2.odf

./Thesis/Chapter3:
[student@desktopX Documents]$
```

Copy files The **cp** command **copies one** or more files to become new, independent files. Syntax allows copying an existing file to a new file in the current or another directory, or copying multiple files into another directory.

- ❑ In any destination, new file names must be unique. If the new file name is not unique, the copy command will overwrite the existing file.

```
[student@desktopX ~]$cd Videos[student@desktopX Videos]$cp blockbuster1.ogg  
blockbuster3.ogg[student@desktopX Videos]$ls -l  
total 0  
-rw-rw-r--. 1 student student 0 Feb 8 16:23 blockbuster1.ogg  
-rw-rw-r--. 1 student student 0 Feb 8 16:24 blockbuster2.ogg  
-rw-rw-r--. 1 student student 0 Feb 8 19:02 blockbuster3.ogg  
drwxrwxr-x. 2 student student 4096 Feb 8 23:35 Watched  
[student@desktopX Videos]$
```

Move files The **mv** command renames files in the same directory, or relocates files to a new directory.

- ❖ File contents remain unchanged. Files moved to a different file system require creating a new file by copying the source file, then deleting the source file. Although normally transparent to the user, large files may take noticeably longer to move.

```
[student@desktopX Videos]$cd ../Documents[student@desktopX Documents]$ls -l
total 0
-rw-rw-r--. 1 student student    0 Feb  8 16:24 thesis_chapter1.odf
-rw-rw-r--. 1 student student    0 Feb  8 16:24 thesis_chapter2.odf
[student@desktopX Documents]$ mv thesis_chapter2.odf thesis_chapter2_reviewed.odf
[student@desktopX Documents]$ mv thesis_chapter1.odf Thesis/Chapter1
[student@desktopX Documents]$ ls -lR
.:
total 16
drwxrwxr-x. 2 student student 4096 Feb 11 11:58 ProjectX
```

Chapter 03

Managing Files from the Command Line

```
drwxrwxr-x. 2 student student 4096 Feb 11 11:55 ProjectY
drwxrwxr-x. 5 student student 4096 Feb 11 11:56 Thesis
-rw-rw-r--. 1 student student  0 Feb 11 11:54 thesis_chapter2_reviewed.odf

./ProjectX:
total 0
-rw-rw-r--. 1 student student 0 Feb 11 11:58 thesis_chapter1.odf
-rw-rw-r--. 1 student student 0 Feb 11 11:58 thesis_chapter2.odf

./ProjectX/Thesis:
total 0

./ProjectY:
total 0

./Thesis:
total 12
drwxrwxr-x. 2 student student 4096 Feb 11 11:59 Chapter1
drwxrwxr-x. 2 student student 4096 Feb 11 11:56 Chapter2
drwxrwxr-x. 2 student student 4096 Feb 11 11:56 Chapter3

./Thesis/Chapter1:
total 0
-rw-rw-r--. 1 student student 0 Feb 11 11:54 thesis_chapter1.odf

./Thesis/Chapter2:
total 0
-rw-rw-r--. 1 student student 0 Feb 11 11:54 thesis_chapter2.odf

./Thesis/Chapter3:
total 0
[student@desktopX Documents]$
```

Managing Files from the Command Line

- ❑ Remove files and directories Default syntax for `rm` deletes files, but not directories. Deleting a directory, and potentially many subdirectories and files below it, requires the `-r` recursive option. There is no command-line undelete feature, nor a trash bin from which to restore.

```
[student@desktopX Documents]$pwd
/home/student/Documents
[student@desktopX Documents]$ rm thesis_chapter2_reviewed.odf
[student@desktopX Documents]$ rm Thesis/Chapter1
rm: cannot remove `Thesis/Chapter1': Is a directory
[student@desktopX Documents]$ rm -r Thesis/Chapter1
[student@desktopX Documents]$ ls -l Thesis
total 8
drwxrwxr-x. 2 student student 4096 Feb 11 12:47 Chapter2
drwxrwxr-x. 2 student student 4096 Feb 11 12:48 Chapter3
[student@desktopX Documents]$ rm -ri Thesis
rm: descend into directory `Thesis'? y
rm: descend into directory `Thesis/Chapter2'? y
rm: remove regular empty file `Thesis/Chapter2/thesis_chapter2.odf'? y
rm: remove directory `Thesis/Chapter2'? y
rm: remove directory `Thesis/Chapter3'? y
rm: remove directory `Thesis'? y
[student@desktopX Documents]$
```

- After `rm` failed to delete directory, the `-r` recursive option succeeded. The last `rm` command parsed into each subdirectory first, individually deleting contained files before removing each now-empty directory. Using `-i` will interactively prompt for each deletion.
- This is essentially the opposite of `-f` which will force the deletion without prompting the user. The `rmdir` command deletes directories only if empty. Removed directories cannot be undeleted.

```
[student@desktopX Documents]$pwd
/home/student/Documents
[student@desktopX Documents]$ rmdir ProjectY
[student@desktopX Documents]$ rmdir ProjectX
rmdir: failed to remove `ProjectX': Directory not empty
[student@desktopX Documents]$ rm -r ProjectX
[student@desktopX Documents]$ ls -lR
.:
total 0
[student@desktopX Documents]$
```



جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY

