

Introduction to Linux Operating System

Lab 01



اللهم علمنا ما ينفعنا ،، وانفعنا بما علمتنا ،، وزدنا علماً



Lab Objective

- To introduce some of the common Linux commands



Introduction

- *Linux* is a clone of the *Unix* operating system.
- Unix was developed in 1969 by Dennis Ritchie and Kevin Thompson at Bell Laboratories.
- Most of the Unix operating system is written in the high-level programming language C.
- A Unix operating system consists of a kernel and a set of common utility programs.
- The kernel is the core of the operating system, which manages the computer hardware, controls program executions, manages memory, etc.
- The utility programs provide user level commands, such as those to create and edit files.



Why Linux?

- Free, **open** source.
- **Ubuntu** is a complete Linux operating system
- At Ubuntu's heart is the Linux kernel
- Ubuntu has a graphical user interface (**GUI**), making it similar to other popular operating systems like Windows and Mac OS
- The OS represents **applications** as icons or menu choices that you can select using keyboard commands or a mouse



How to Use Ubuntu?

- Requirements:

1- **VirtualBox** (on your Windows or Mac computer)

<https://www.wikihow.com/Install-VirtualBox>

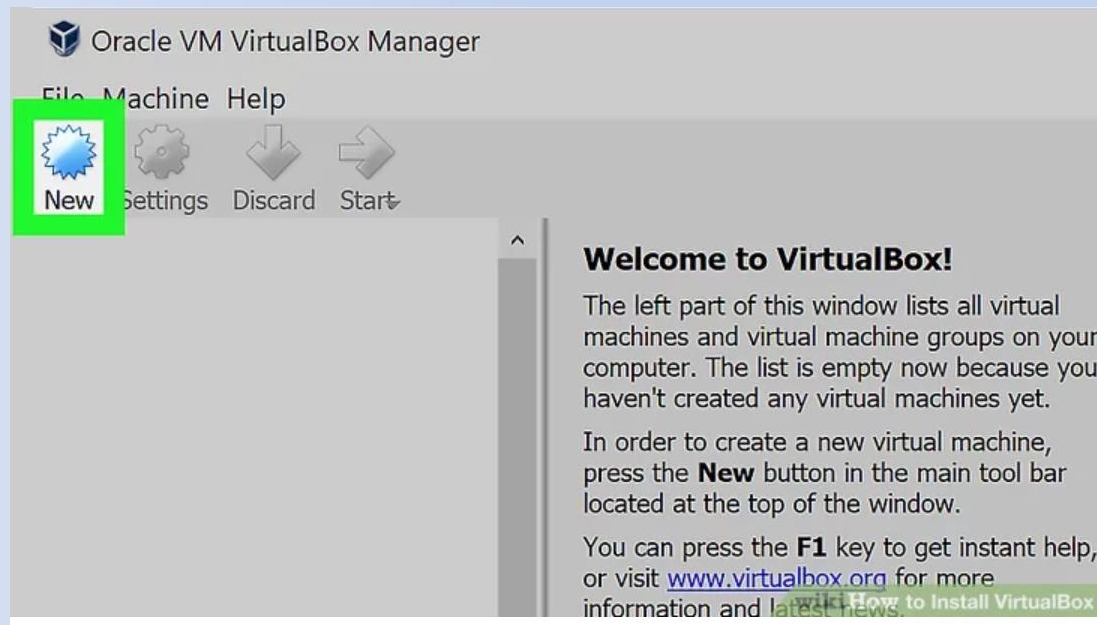
2- **Ubuntu disk image** (ISO File)

<https://ubuntu.com/download/desktop>



How to Use Ubuntu?

- Once you have download the VirtualBox:
 - 1- **Install** the **Ubuntu** operating system by using its **ISO** file on the Virtual Machine.
 - 2- Open **VirtualBox** and click on New tab.



How to Use Ubuntu?


3- Identify the operating system as following:

Name and operating system

Please choose a descriptive name and destination folder for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

Name:

Machine Folder:

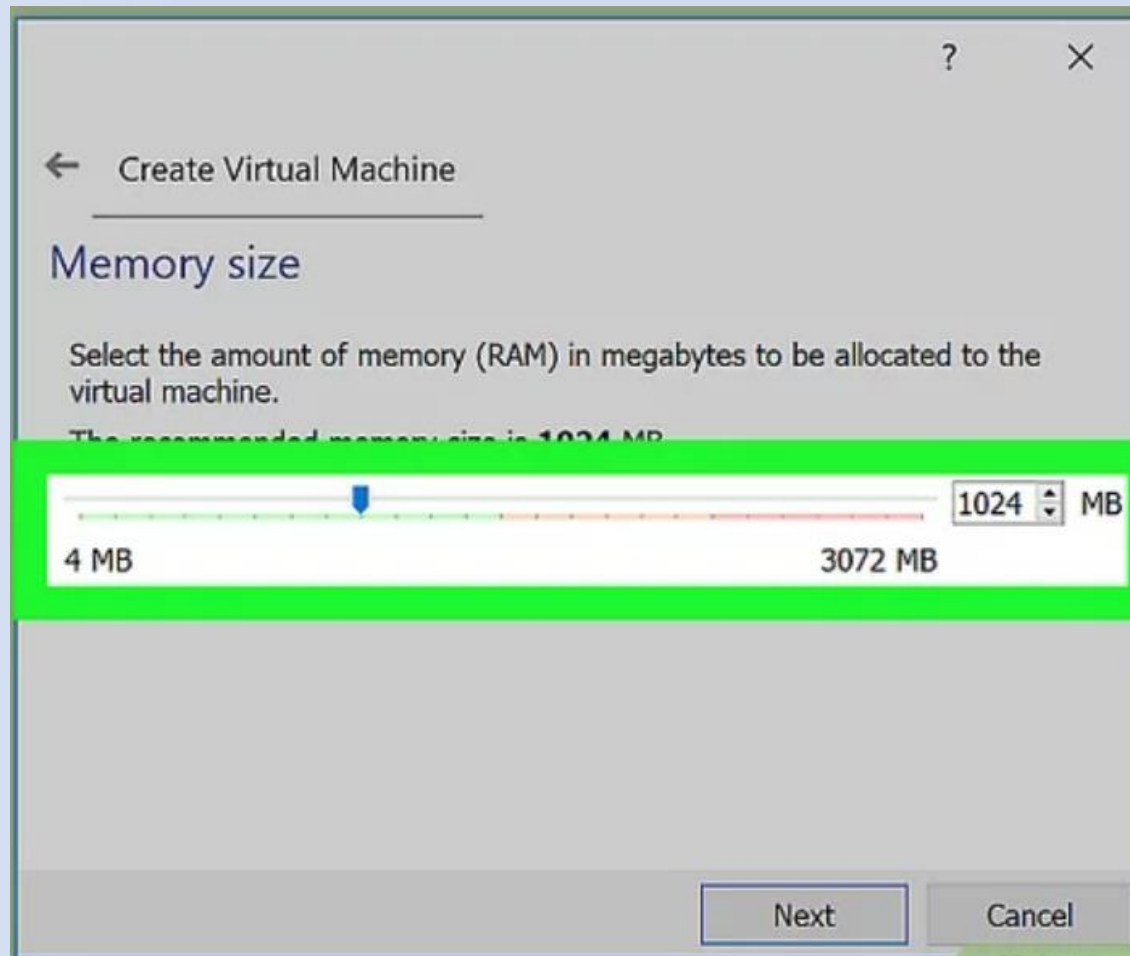
Type: 

Version:



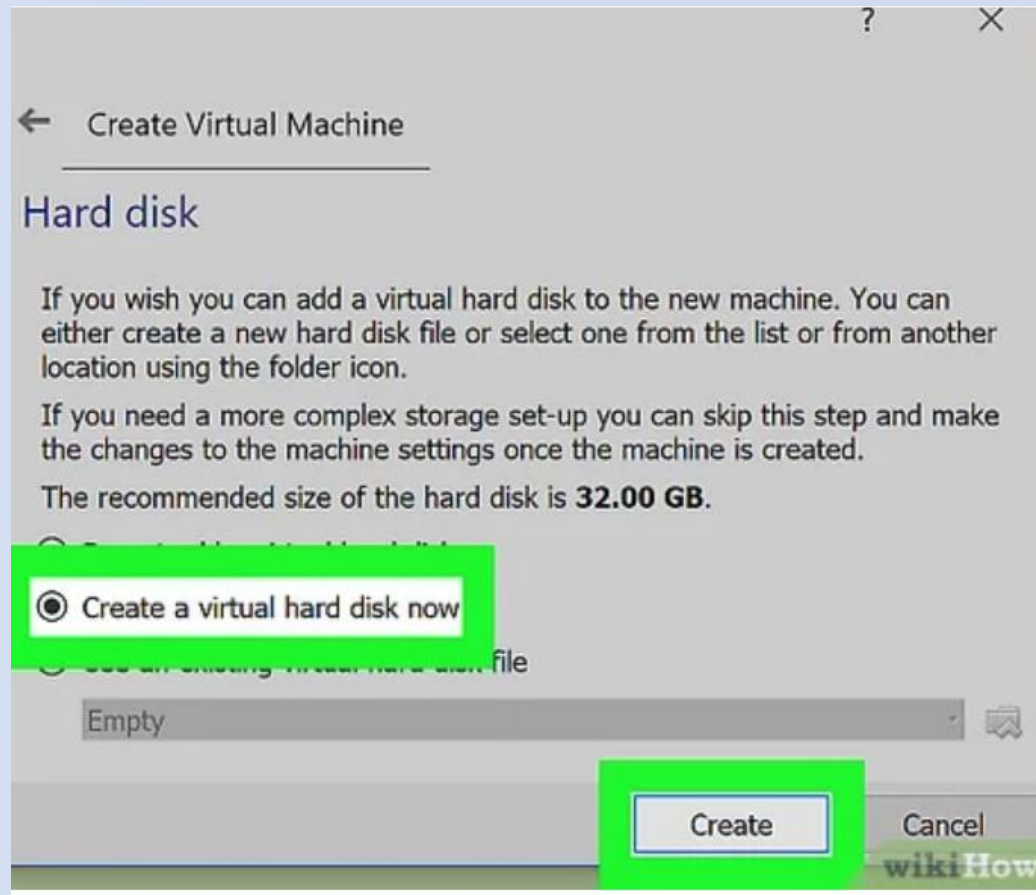
How to Use Ubuntu?

4- Set the amount of RAM as following:



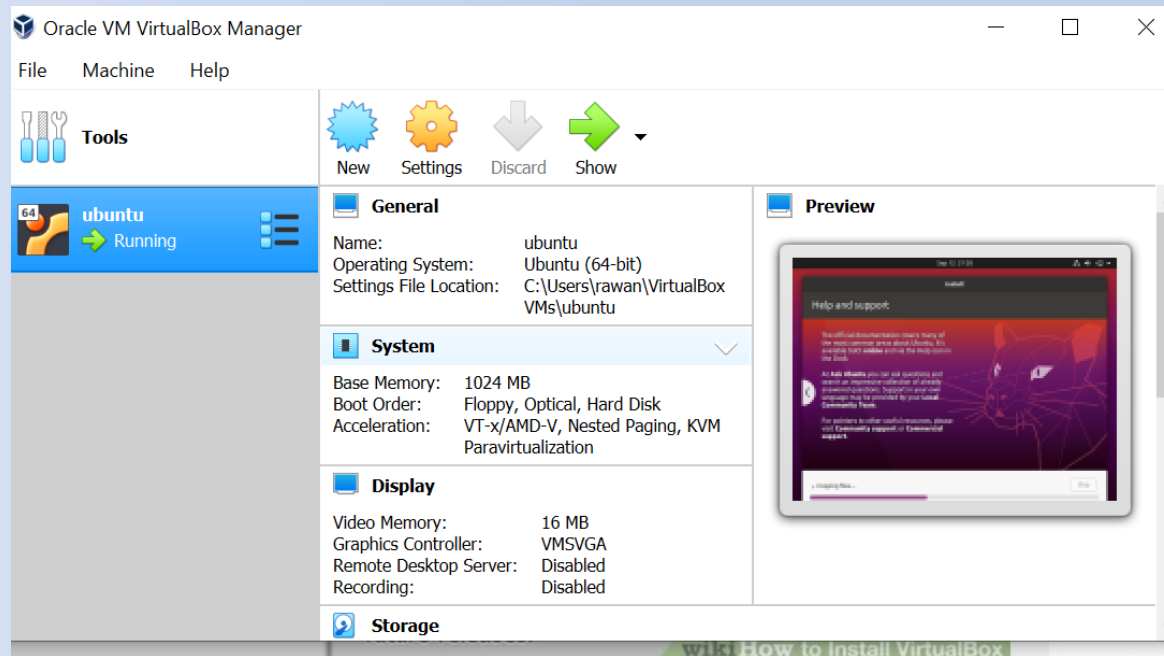
How to Use Ubuntu?

5- Create a virtual hard drive as following:



How to Use Ubuntu?

6- Once the virtual machine has been configured, **Start the operating system installation**. Double-click your new machine (ubuntu) in the left menu, then browse through your computer for the installation image file



How to Use Ubuntu?

7- Click Start to prompt VirtualBox to begin reading your ISO file.



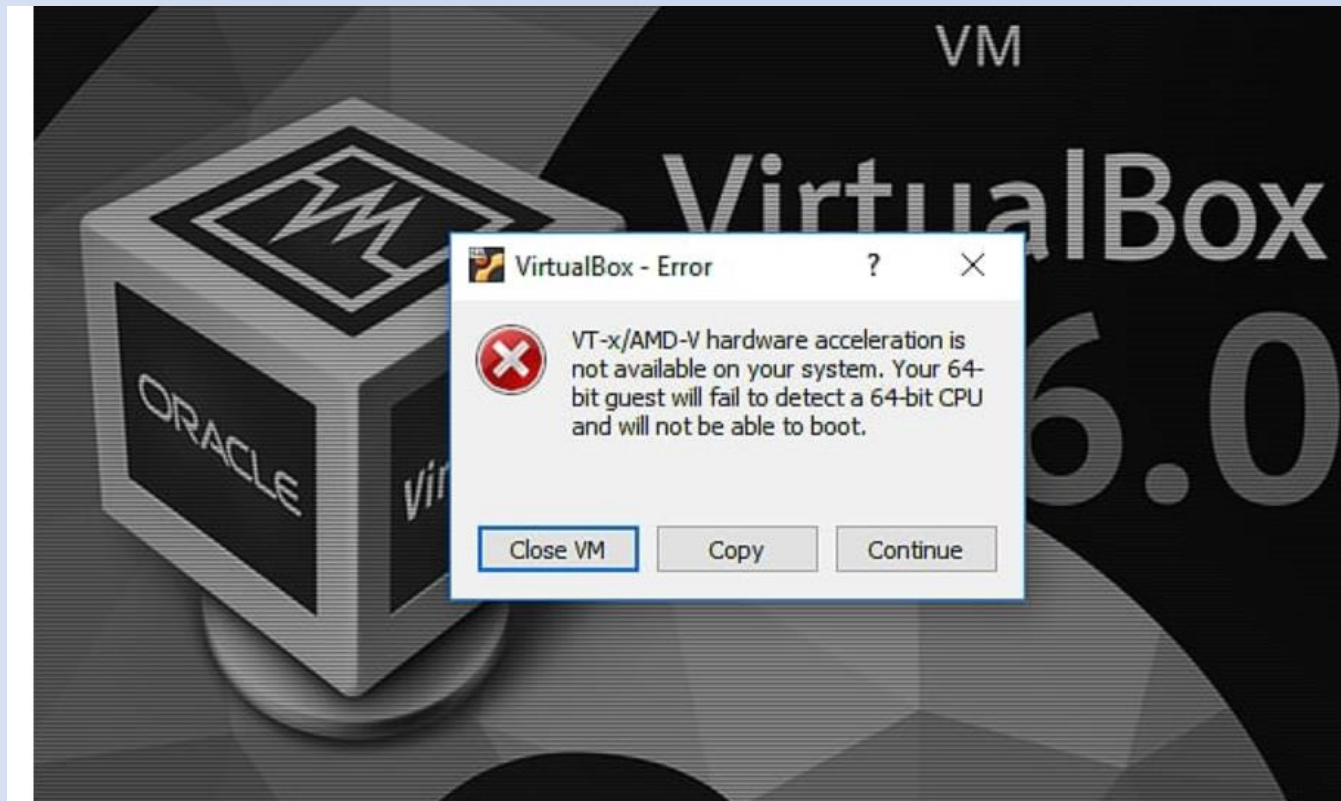
How to Use Ubuntu?

8- **Boot up your virtual machine.** Once the operating system is installed, your virtual machine is ready to go. Simply double-click the name of your virtual machine in the left menu of the VirtualBox main page to start it up.



How to Use Ubuntu?

You may encounter an ERROR when you try to start up Ubuntu as this message shows:



How to Use Ubuntu?

To solve that problem, you have to **enable Virtualization by Restarting your computer and booting. HOW?**

- By pressing F12 or F2 while starting the system to get the **booting menu**, then **choose advanced setting and enable Virtualization or choose (VT-X / AMD V) from Virtualization menu.**
- **Save the changes and exit.**



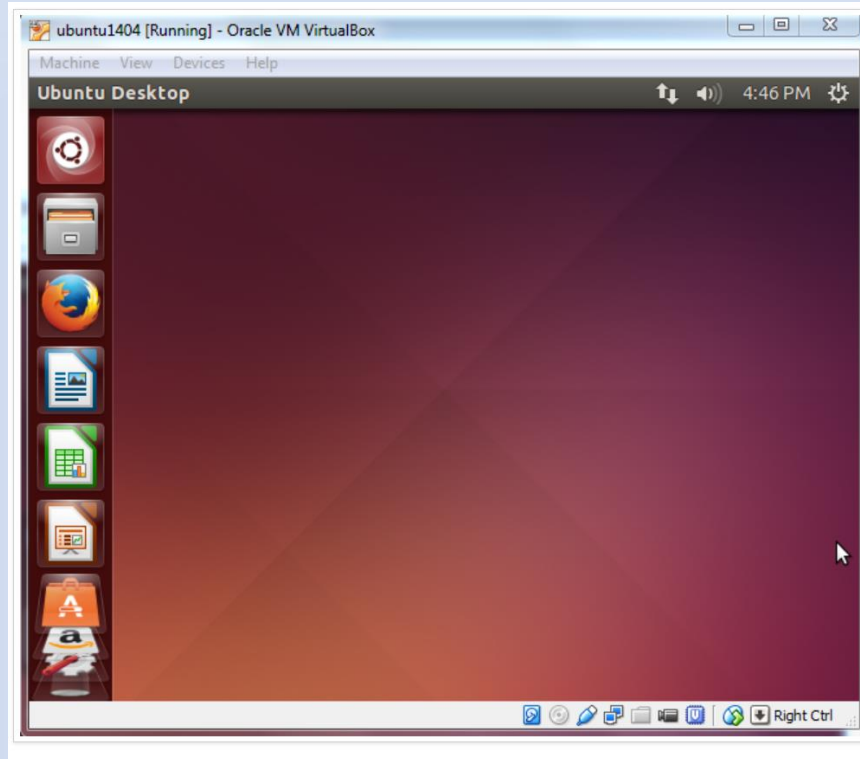
How to Use Ubuntu?

Once there is no problem, Install Ubuntu as described in the following link.

<https://brb.nci.nih.gov/seqtools/installUbuntu.html#install>

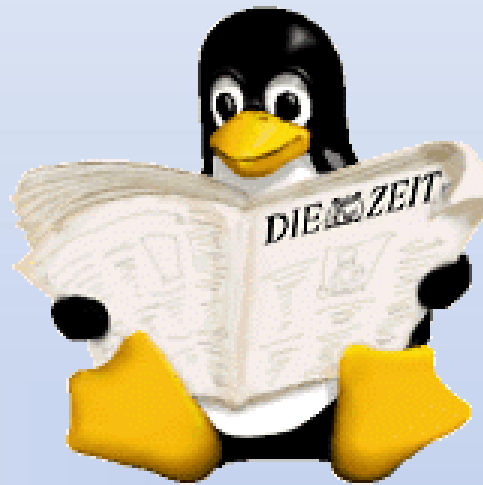


Ubuntu



- This screenshot shows the Ubuntu desktop. A Web browser opens by default. You can minimize or close it to get it out of the way.





LINUX COMMANDS OVERVIEW



Starting an UNIX Terminal

- To open an UNIX terminal window, click on the "Terminal" icon in the lunch bar.



- An UNIX Terminal window will then appear with a **\$** prompt, waiting for you to start entering commands.
- Unix Terminal is like Windows DOS



General Linux Command Format

- A little like DOS commands on windows with some differences

The Command

```
$ cmd -[option(s)] [argument(s)]
```

One or more the directory/file to apply the command to

One or more options to change the behavior of the command

- Notes:
 - Parts between [] packets are optional
 - Linux is CASE SENSITIVE



Getting Help

- In Linux, there are on-line manuals which gives information about most commands.
- `man` is used to read the manual page for a particular command one page at a time:

```
$ man cmd
```

- **Examples,**

```
$ man ls
```



Displays the manual pages of the command **ls**

```
$ man man
```



Displays the manual pages of the command **man**

- Use the following keys to go through the manual
 - Enter → one line forward
 - F → Forward one window OR
 - B → Backward one window OR screen
 - Q → Quits the manual



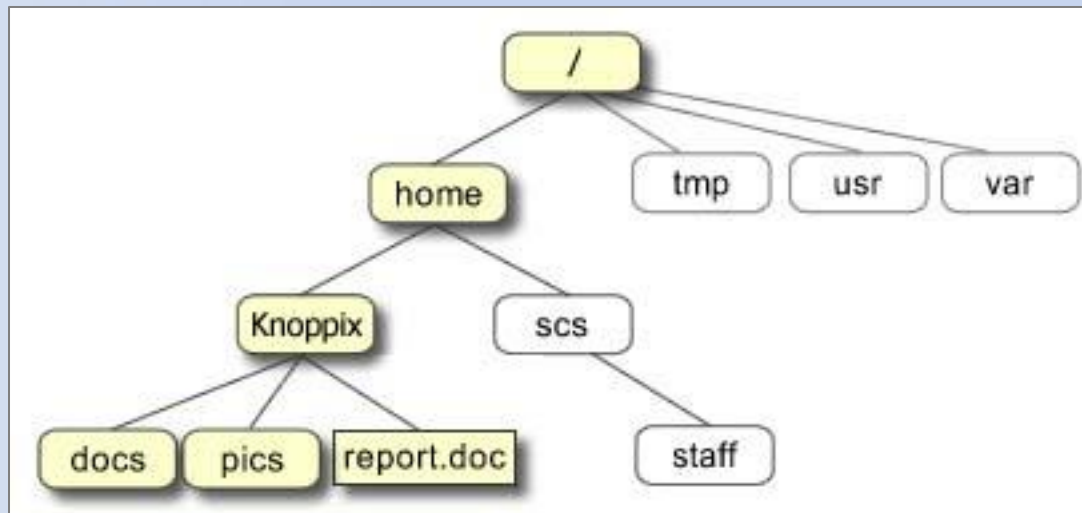


DIRECTORY COMMANDS



What is a Directory?

- In Linux, all the files are grouped together in the directory structure.
- The file-system is arranged in a hierarchical structure, like an inverted tree.
- The top of the hierarchy is called **root** (written as a slash **/**)



- In the diagram above, the full path to the file **report.doc** is:
/home/knoppix/report.doc



Pathnames

- **pwd** (print working directory) is used to prints the current directory, type:

```
$ pwd
```



The full pathname will look something like this:
/home/rawan



Making and Removing Directories

- **mkdir** and **rmdir** are used for making and removing directories.

```
$ mkdir dirname
```

→ Creates a new directory with name ***dirname*** in the current directory

```
$ rmdir dirname
```

→ Deletes the directory ***dirname*** from the current directory

- **Note:** A directory must not contain any files when it is deleted, otherwise an error message is displayed.

- Examples:

```
$ mkdir dir1
```

→ Creates a new directory called **dir1**

```
$ rmdir dir3
```

→ Removes the directory **dir3** (if it exists)



Changing to a Different Directory

- **cd** (Change Directory) is used to change the working directory.

```
$ cd dirpath
```

→ Changes the current directory to the relative or absolute pathname of the directory ***dirpath***.

```
$ cd
```

→ If no directory is given, the command changes the current directory to the home directory.

```
$ cd ..
```

→ Changes to the parent directory.

- Examples:

```
$ cd
```

→ Change to home-directory

```
$ cd dir1
```

→ Change to directory **dir1**

```
$ cd dir2
```

→ Error because **dir2** is not in **dir1**

```
$ cd ..
```

→ Change to parent directory **dir1**

```
$ cd dir2
```

→ Change to directory **dir2**

```
$ cd /home/knoppix/dir1
```

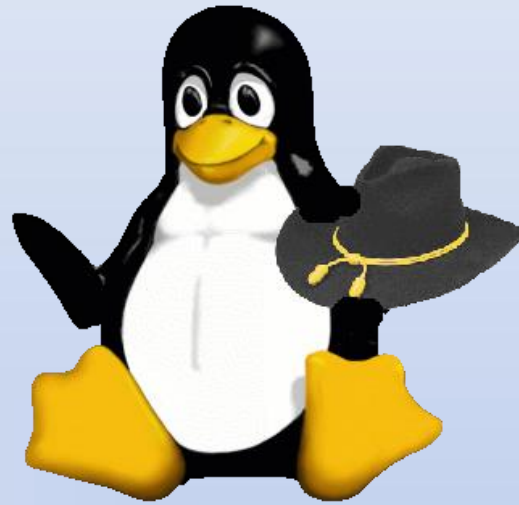
→ Change to directory **dir1**



Directory Commands Summary

Command	Meaning
pwd	display the path of the current directory
mkdir <i>dirname</i>	make a directory
rmdir <i>dirname</i>	remove a directory
cd <i>directory</i>	change to named directory
cd	change to home-directory
cd ..	change to parent directory





FILE COMMANDS



What is a file?

- A file is a collection of data.
- They are created by users using text editors, running compilers etc.
- Examples of files:
 - a document (report, essay etc.)
 - the text of a program written in some high-level programming language (like C or C++)



Listing files and directories

- **ls** (list) is used to list information about files and directories.

```
$ ls dirpath
```

→ If the command has a directory name as argument (i.e., *dirpath*), then the command lists the files in that directory.

```
$ ls
```

→ If no directory is given, then the command lists the files in the current directory.

```
$ ls -l
```

→ Includes extensive information on each file.

- Note: The **ls** command has several options. The most important is **ls -l**, which includes extensive information on each file, including, the access permissions, owner, file size, and the time when the file was last modified.



Moving and renaming Files

- **mv** is used to rename or move a file or a directory.

```
$ mv fname newfile
```

→ The file or directory **fname** is renamed as **newfile**. If the destination file (**newfile**) exists, then the content of the file is overwritten, and the old content of **newfile** is lost.

```
$ mv fname dirname
```

→ If the first argument is a file name and the second argument is a directory name (**dirname**), the file is moved to the specified directory.

- Examples:

```
$ mv dir2 dir5
```

→ Renames **dir2** to **dir5**

```
$ mv dir5 dir1
```

→ Moves **dir5** to **dir1**

```
$ mv file2 dir1
```

→ Moves **file1.txt** to **dir1**



Copying and Removing Files

- **cp** (copy) and **rm** (remove) are used to copy and remove files:

```
$ cp fname newfile
```

→ Copies the content of file *fname* to *newfile*. If a file with name *newfile* exists the content of that file is overwritten.

```
$ cp fname dirname
```

→ If the second argument is a directory, then a copy of *fname* is created in directory *dirname*.

```
$ rm fname
```

→ Removes the file *fname* from the current directory

- Examples:

```
$ cp file1 dir1
```

→ Copy **file1** to **dir1**

```
$ cd dir1
```

```
$ cp file1 file2
```

→ Copy to **file1** to **file2** overwriting its content

```
$ rm file1
```

→ Removes **file1**



View and Modify Text Files

- **more** and **cat** are used to view and modify text files.

```
$ more fname
```

→ Displays the contents of file **fname**, one page at a time.

```
$ cat fname
```

→ Similar to the more command, but the file is displayed without stopping at the end of each page

- Examples:

```
$ more file1
```

→ Displays the contents of **file1**

```
$ cat file1
```

→ Displays the contents of **file1**



File Commands Summary

Command	Meaning
ls	list files and directories in the current directory
ls <i>dirpath</i>	List files and directories in <i>dirpath</i>
ls -l	Includes extensive information on each file
mv <i>file1</i> <i>file2</i>	rename <i>file1</i> to <i>file2</i>
mv <i>file1</i> <i>dirpath</i>	move <i>file1</i> to <i>dirpath</i>
cp <i>file1</i> <i>file2</i>	copy <i>file1</i> and call it <i>file2</i>
cp <i>file1</i> <i>dirpath</i>	copy <i>file1</i> to <i>dirpath</i>
rm <i>file</i>	remove a file
more <i>file</i>	display a file
cat <i>file</i>	display a file



Redirecting Programs Output

- `>` and `>>` are used to redirect program output

```
$ cmd > fname
```

→ The output of ***cmd*** is written to file ***fname***. The file is created if it doesn't already exist, and the contents is overwritten if the file exists.

```
$ cmd >> fname
```

→ Appends the output of command ***cmd*** to the end of file ***fname***.

- Examples:

```
$ ls > mylist
```

→ Writes a listing of the current directory in file ***mylist***

```
$ ls >> mylist
```

→ Appends a listing of the current directory to file ***mylist***





PROCESSES AND JOBS COMMANDS



Foreground and Background Processes

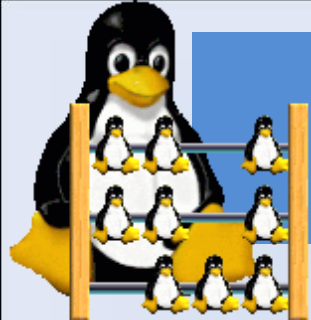
- A *process* is an executing program identified by a unique PID (process identifier).
- In Linux, each terminal window can run multiple commands at the same time.
- It is possible to stop a command temporarily and resume it at a later time.
- In each terminal window, one command can be run as a *foreground* process and multiple command can be run as *background* processes.



Processes and Jobs Commands

Command	Meaning
Ctrl+C	Terminates the command running in the foreground
Ctrl+Z	Stops (suspend) the commands in the foreground.
cmd&	Executes the command <i>cmd</i> in the background
bg	background the suspended job
jobs	Lists all background and stopped commands of the current user, and assigns a number to each command.
fg %n	Resume suspended job number n in the foreground, and make it the current job. The numbers are as displayed by the jobs command.
bg %n	Resumes suspended job number n in the background, as if it had been started with &.
ps -all	Lists all current processes and their assigned ID (pid)
kill pid	Terminates the process with the specified ID: <i>pid</i> , where <i>pid</i> is as displayed by the command <i>ps</i>





Exercise

- List all the content of the home directory then remove any subdirectory in it
- Go to the home directory then make 3 new subdirectory called (**pics**, **docs**, **backup**)
- Make a subdirectory in (**pics**), call it (**babies**)
- Rename the (**backup**) directory to (**bup**) then move it to the (**docs**) directory
- Write a listing of the current directory in a file called (*list_a*)
- Copy the file (*list_a*) to the (**docs**) directory
- Make a copy of (*list_a*) and call it (*list_b*) then move (*list_b*) to (**bup**) directory
- Run the command that displays the manual of the (passwd) command in the background
- Terminate all the background process



??? ANY QUESTIONS ???

