LINUX TUTORIALS

Dr. Pheneas Nkundabakura

University of Rwanda –College of Education

Session 1: 14:30-16:00

Session 2: 16:30-18:00

ASP2016 – KIGALI

Kigali, 2 August 2016

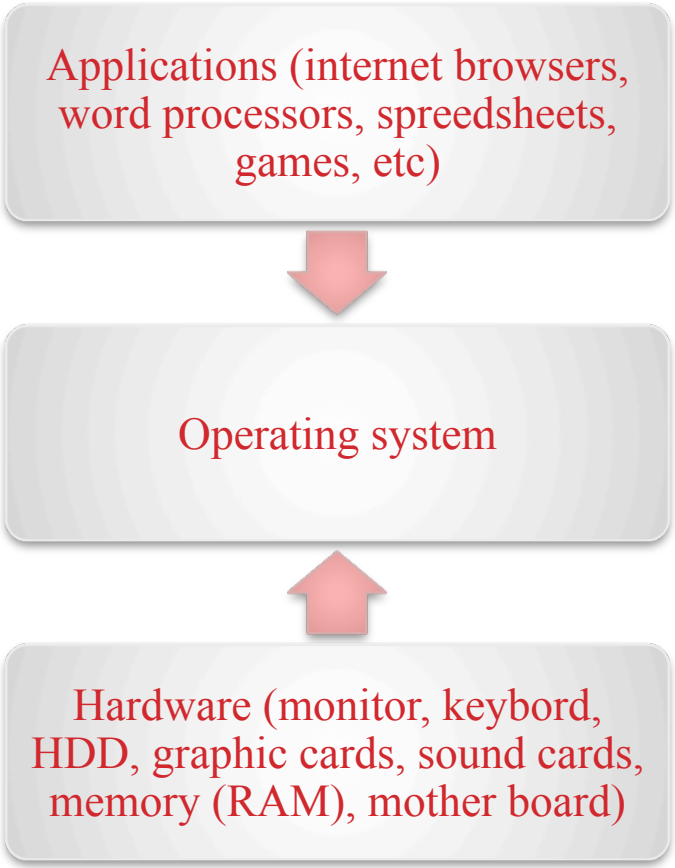
UNIVERSITY OF RWANDA

Contents

- Introduction to Linux?
- Shell or Terminal
- Getting started
- Linux directory hierarchy
- Linux Commands
- Tutorials

What is an Operating System (OS)?

An operating system is the software that provides the interface between the hardware of a computer system and the applications/ programs that are used on it.



Tasks of an Operating System?

- Control Hardware
- Run Applications
- Manage Data and Files

What are the different types?

Linux is a Unix-like computer operating system assembled under the model of free and open source software development and distribution.

Microsoft Windows is a series of graphical interface operating systems developed, marketed, and sold by Microsoft.

Mac OS is a series of graphical user interface-based operating systems developed by Apple Inc. for their Macintosh

iOS (previously **iPhone OS**) is a mobile operating system developed and distributed by Apple Inc. Originally unveiled in 2007 for the iPhone, it has been extended to support other Apple devices such as the iPod Touch

Android is a Linux-based operating system designed primarily for touchscreen mobile devices such as smartphones and tablet computers. Initially developed by Android, Inc.

BSD/OS had a reputation for reliability in server roles; the renowned Unix programmer and author W. Richard Stevens used it for his own personal web server for this reason.

Advantages of Linux?

- A Unix-like Operating System
- Multi-user, Multitasking, Multiprocessor
- Has the X Windows GUI
- Coexists with other Operating Systems
- Runs on multiple platforms
- Includes the Source Code

- Easy to install applications.
- Secure
- Easy to change Options
- Community
- Free

Who invented Linux?

- Linux was created by **Linus Torvalds (Finland) in 1991** with assistance from programmers around the world

Today Linux is used on 7-10 million computers with 1000's of programmers working to enhance it

Linux is free software

To qualify as free software by FSF (Free Software Foundation) standards, you must be able to:

- Run the program for any purpose you want to, rather than be restricted in what you can use it for.
- View the program's source code.
- Study the program's source code and modify it if you need to.
- Share the program with others.
- Improve the program and release those improvements so that others can use them.

Advantage to be an Open Source Software

- When programmers on the Internet can **read**, **redistribute**, **modify, improve, adapt** the source, the software **evolves**

.

Popular Linux Distributions

- SUSE
- Fedora
- Red Hat Enterprise
- Debian
- CentOS
- ALT
- Gentoo
- Turbo
- Mandrake
- Linspire
- Slackware
- **Ubuntu**

Linux distributions (Cont.)

- A great table providing an overview and comparison of most Linux distributions is available at **http://en.wikipedia.org/wiki/Comparison_of_Linux_distributions**

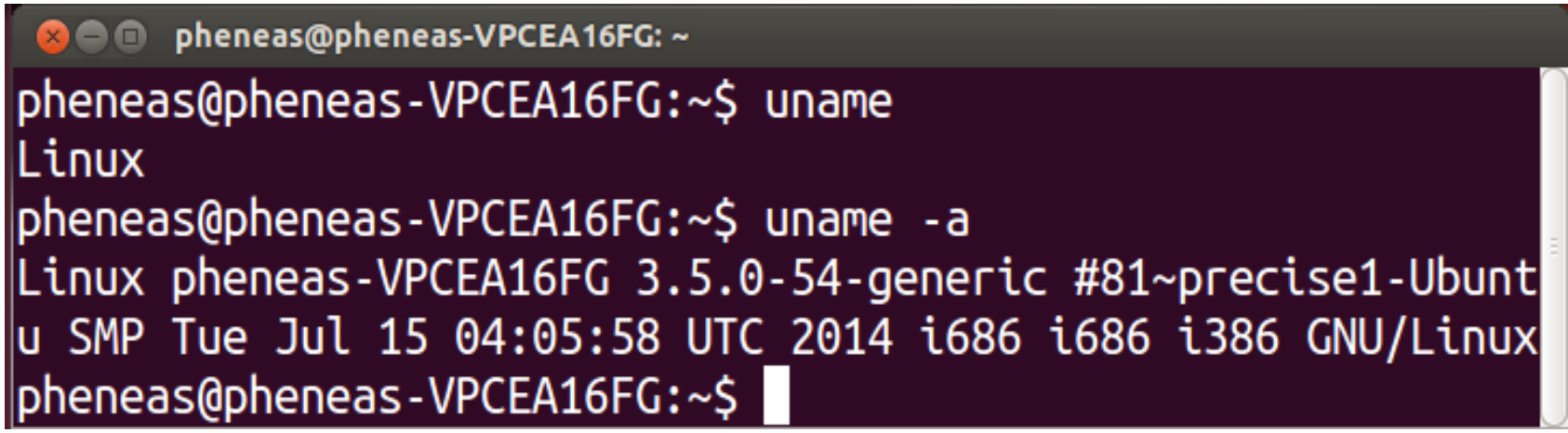
Which Linux Distribution are you running?

In order to know the Lunix Distribution you are running type:

$: uname -a or cat /etc/issue

or cat /proc/version

(we will see shortly about commands)



Shell or Terminal

- An interface between the Linux system and the user
- Used to call commands and programs
- An interpreter
- Equivalent to **cmd** in Windows

Another definition of a Shell

- A shell is any program that takes input from the user (keyboard, mouse), translates it into instructions that the operating system can understand, and conveys the operating system's output back to the user.
  - i.e. Any User Interface
  - Character Based v Graphics Based

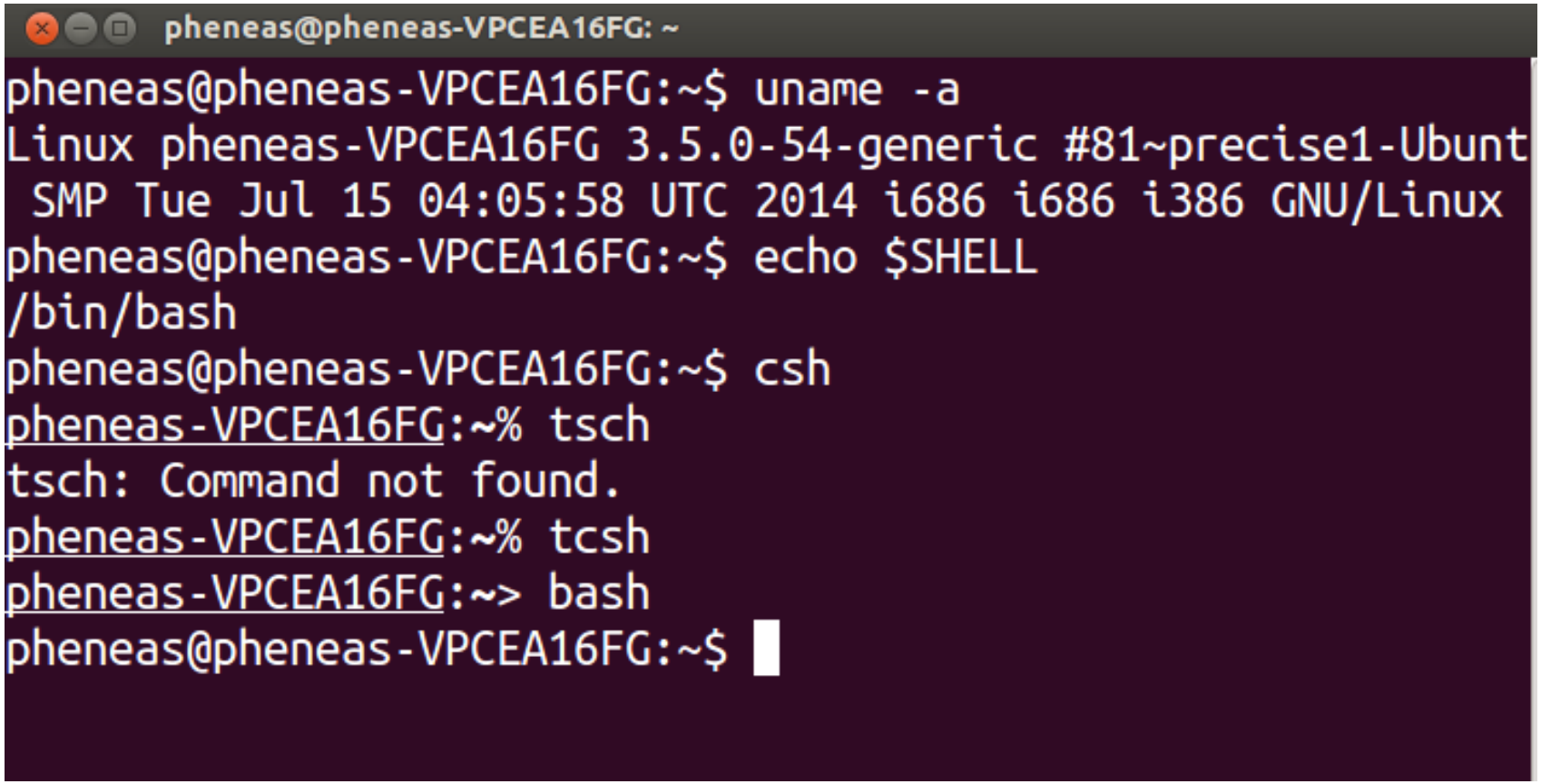
Shell Types

- sh – Bourne shell
- csh – C shell
- ksh – Korn shell
- bash – Bourne Again (Bash) shell
- tcsh – TENEX C shell
- Zch –Z shell

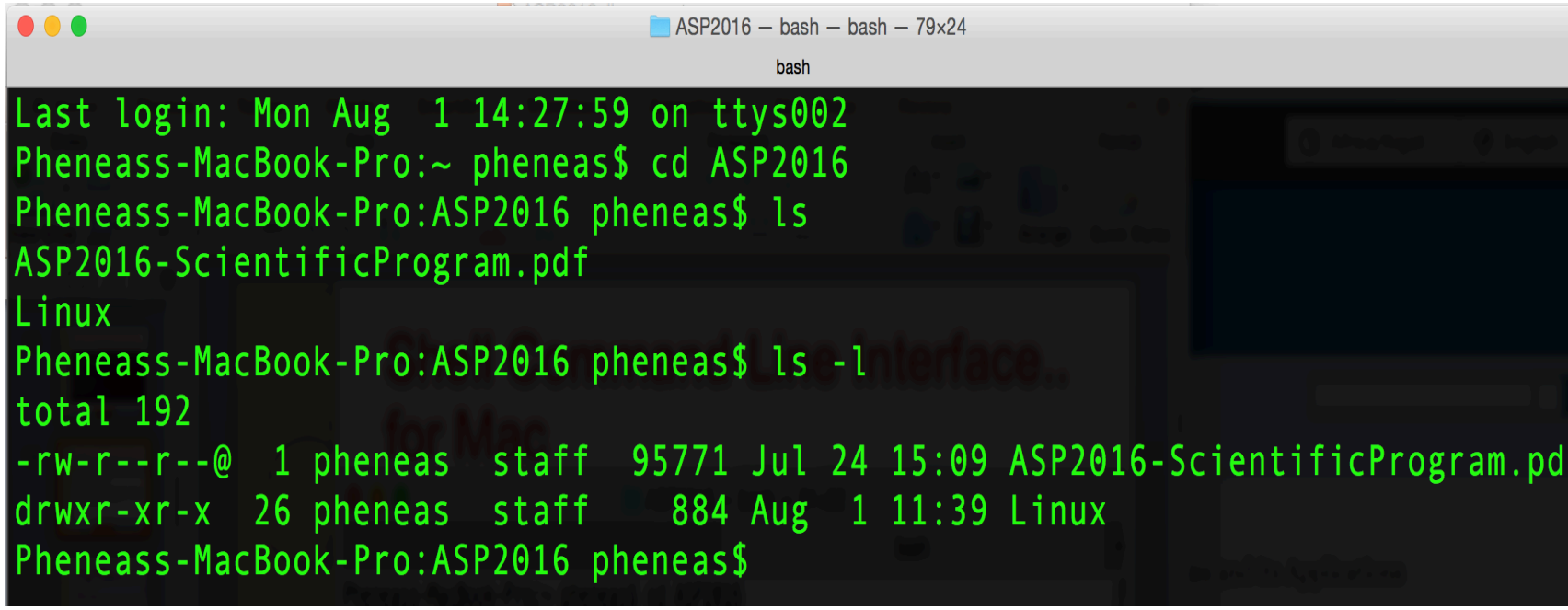
etc

- bash (default for Linux) and csh are the most common
- Shells can be changed by simply typing the name of the shell at the command prompt
- To know the type of the shell: echo $SHELL

Shell Command-Line Interface..



Shell Command-Line Interface.. for Mac



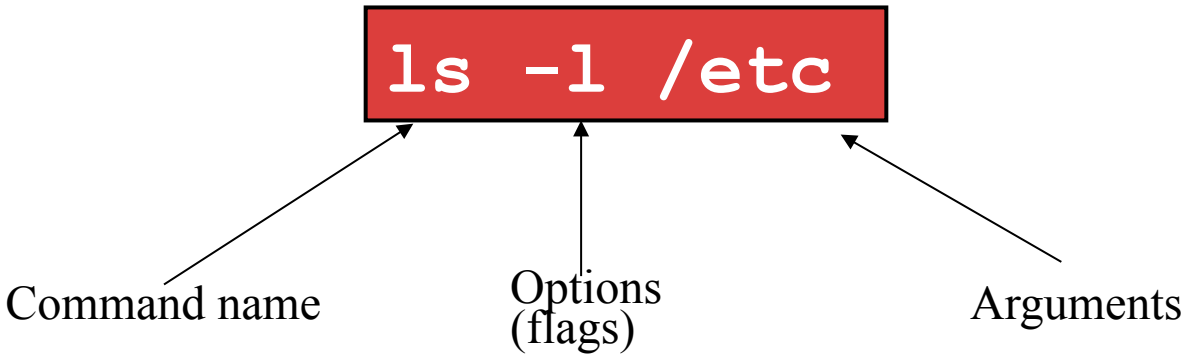
Linux Commands

A *command* is an instruction given by a user telling a computer to do something, such as run a single *program* or a group of linked programs.

Linux Commands

A *command* is an instruction given by a user telling a computer to do something, such as run a single *program* or a group of linked programs.

Example:



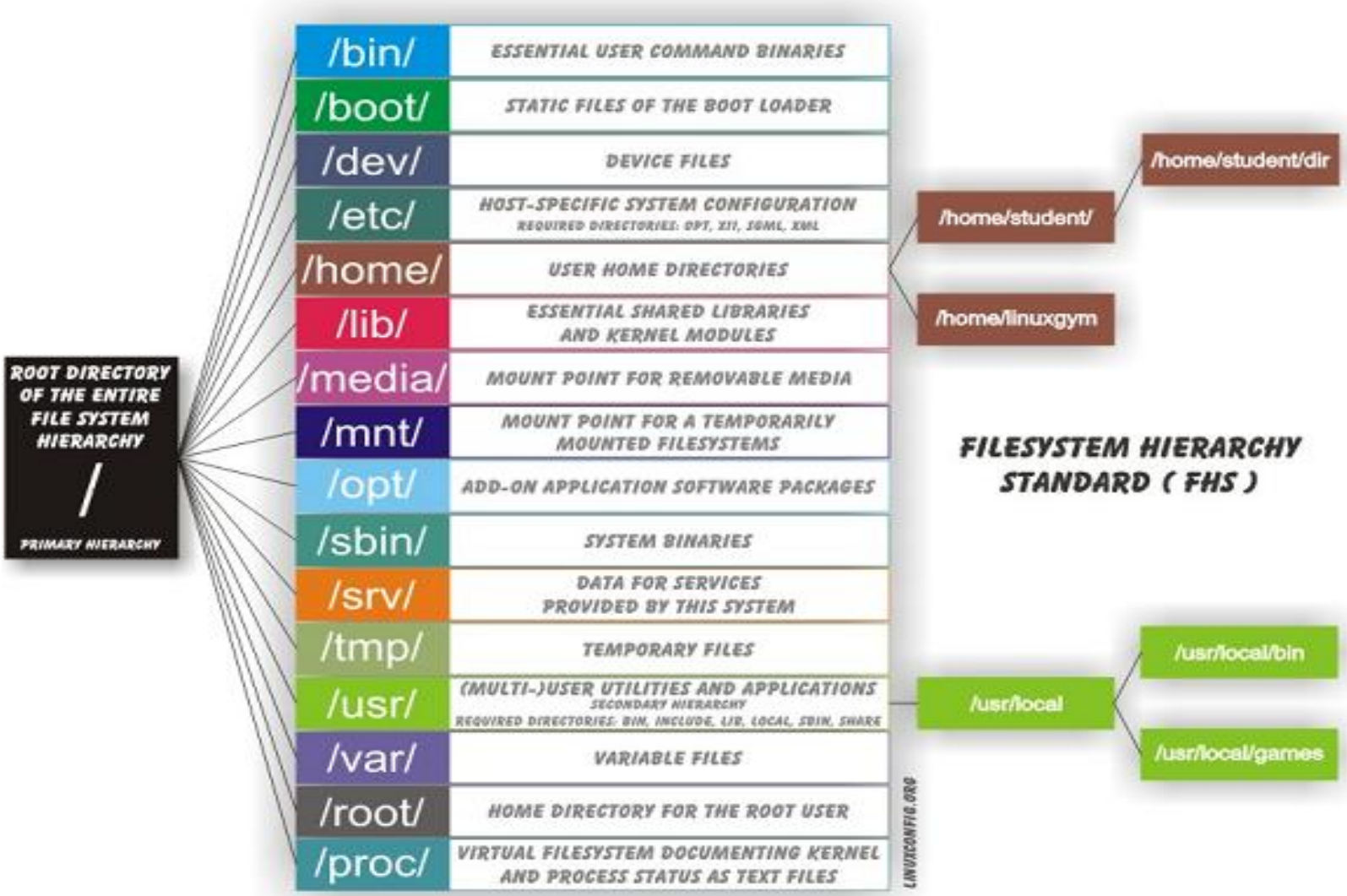
Linux help

- The Linux equivalent of Windows's HELP is **`man`** (manual)
- Use **`man <command>`** to display help for that command
  - Output is presented as a page at a time. Use **`b`** for to scroll backward, **`f`** or a space to scroll forward and **`q`** to quit

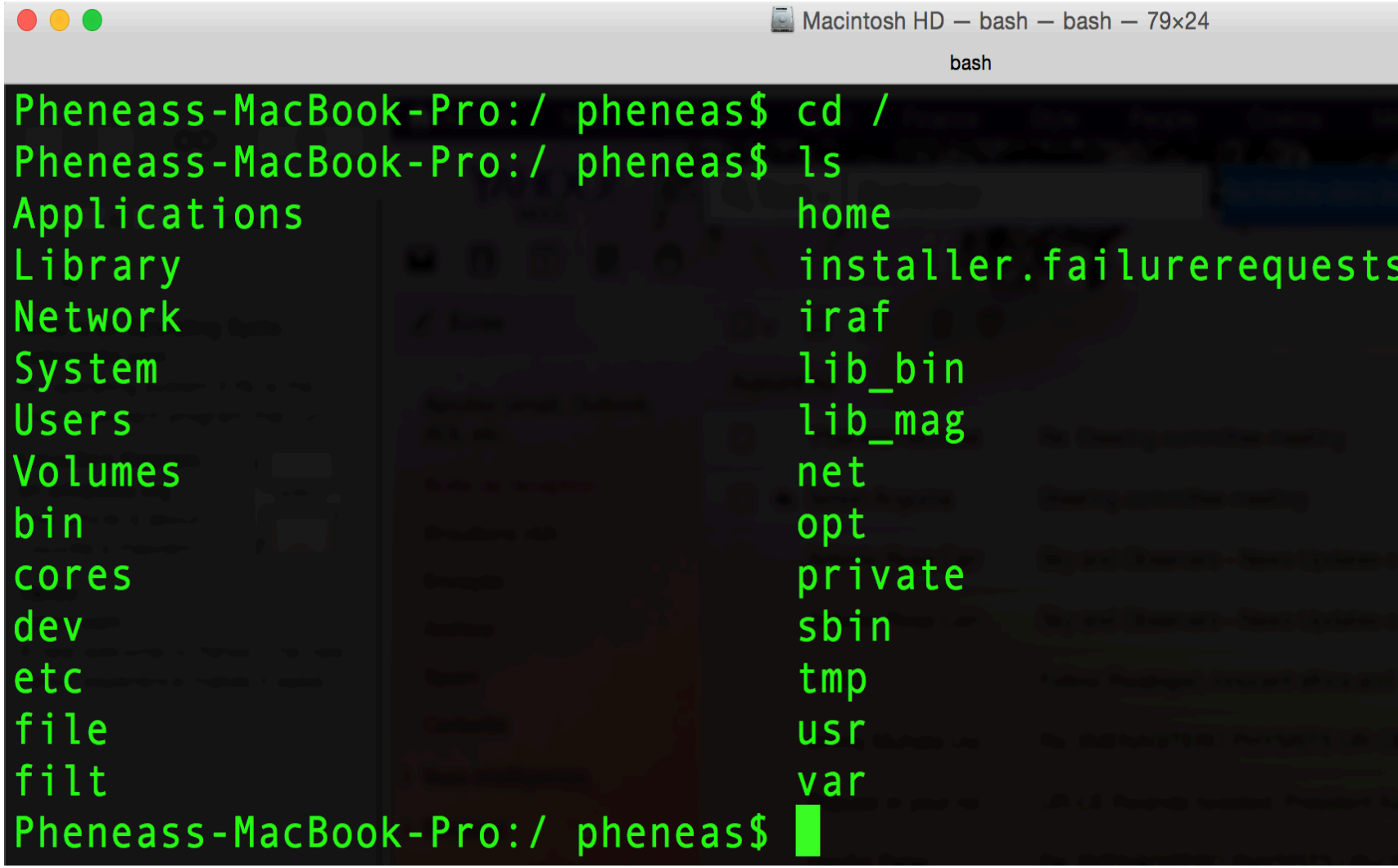
Getting started – Logging in

- Before you can use a computer you must login by specifying your account (username) and password.

Linux Directory Hierarchy



Linux Directory Hierarchy (Mac)



Subdirectories inside the root directory

- **/bin** : Important Linux commands available to the average user.
- **/boot** : The files necessary for the system to boot. Not all Linux distributions use this one. Fedora does.
- **/dev** : All device drivers. Device drivers are the files that your Linux system uses to talk to your hardware. For example, there's a file in the /dev directory for your particular make and model of monitor, and all of your Linux computer's communications with the monitor go through that file.
- **/etc** : System configuration files.

- **/home** : Every user except root gets her own folder in here, named for her login account. So, the user who logs in with chantal has the directory /home/chantal, where all of her personal files are kept.
- **/lib** : System libraries. Libraries are just bunches of programming code that the programs on your system use to get things done.

Subdirectories inside the root directory:

- **/mnt :** Mount points. When you temporarily load the contents of a CD-ROM or USB drive, you typically use a special name under /mnt. For example, many distributions (including Fedora) come, by default, with the directory /mnt/cdrom, which is where your CD-ROM drive's contents are made accessible.
- **/root** : The root user's home directory.
- **/sbin** : Essential commands that are only for the system administrator.
- **/tmp** : Temporary files and storage space. Don't put anything in here that you want to keep. Most Linux distributions (including Fedora) are set up to delete any file that's been in this directory longer than three days.
- **/usr** : Programs and data that can be shared across many systems and don't need to be changed.
- **/var** : Data that changes constantly (log files that contain information about what's happening on your system, data on its way

Tutorial One:

- *Listing files and directories*
- *Making directories*
- *Changing to a different directory*
- *The directories . and ..*
- *Pathnames*
- *More about home directories and pathnames*

Tutorial One:

`ls`	list files and directories
`ls -a`	list all files and directories
`mkdir`	make a directory
`cd directory`	change to named directory
`cd`	change to home-directory
`cd ~`	change to home-directory
`cd ..`	change to parent directory
`pwd`	display the path of the current directory

Review Tutorial one: Special directories and files

- Some file names are special:
  - **/** The root directory (not to be confused with the root user)
  - **.** The current directory
  - **..** The parent (previous) directory
  - **~** My home directory
- Examples:
  - **./file1** same as `file1`
  - **../stuff/x** go up one level then look in directory stuff for `x`

Review Tutorial one: Special directories and Files

- **/home** - all users' home directories are stored here but on Mac all users are in **/Users**
- **/bin, /usr/bin** - system commands
- **/sbin, /usr/sbin** - commands used by sysadmins
- **/etc** - all sorts of configuration files
- **/var** - logs, spool directories etc.
- **/dev** - device files
- **/proc** - special system files

Tutorial Two:

- Copying Files
- Moving Files
- Removing Files and directories
- Displaying the contents of a file on the screen
- Searching the contents of a file

Tutorial Two:

`cp file1 file2`	copy file1 and call it file2
`mv file1 file2`	move or rename file1 to file2
`rm file`	remove a file
`rmdir directory`	remove a directory
`cat file`	display a file
`more file`	display a file a page at a time
`head file`	display the first few lines of a file
`tail file`	display the last few lines of a file
`grep 'keyword' file`	search a file for keywords
`wc file`	count number of lines/words/characters in file

Tutorial Three

- Redirection
- Redirecting the Output
- Redirecting the Input
- Pipes

Tutorial Three

`command > file`	redirect standard output to a file
`command >> file`	append standard output to a file
`command < file`	redirect standard input from a file
`command1 | command2`	pipe the output of command1 to the input of command2
`cat file1 file2 > file0`	concatenate file1 and file2 to file0
`sort`	sort data
`who`	list users currently logged in
`a2ps -Pprinter textfile`	print text file to named printer
`lpr -Pprinter psfile`	print postscript file to named printer

Redirecting Output

- The output of a command may be sent (piped) to a file:

```
ls -l >output
```

"<" is used to specify the output file

Redirecting Input

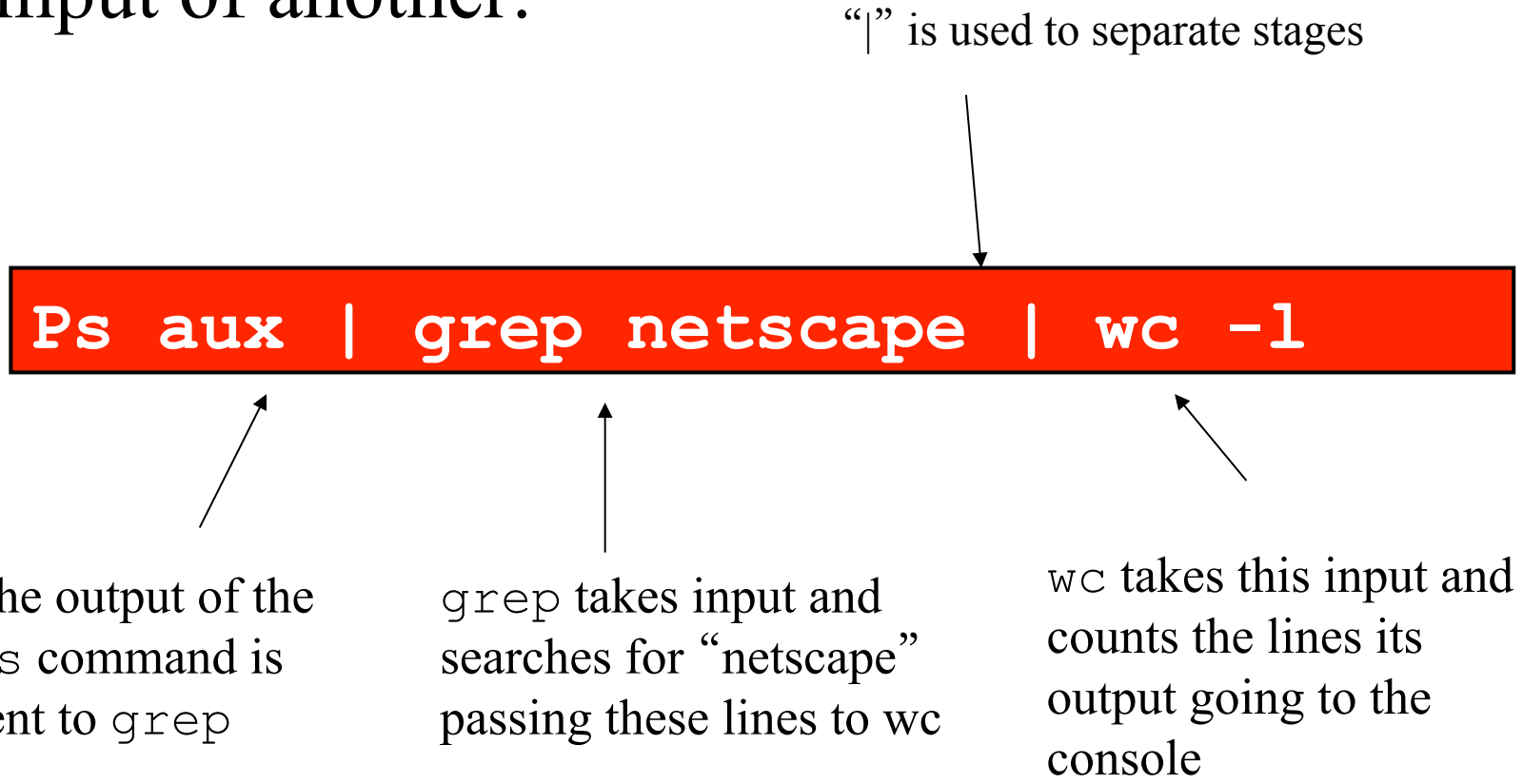
- The input of a command may come (be piped) from a file:

```
wc <input
```

"<" is used to specify the input file

Connecting commands with Pipes

- The output of one command can become the input of another:



Tutorial Four

- **Wildcards**
- **Filename conventions**
- **Getting help**

`*`	match any number of characters
`?`	match one character
`man command`	read the online manual page for a command
`whatis command`	brief description of a command
`apropos keyword`	match commands with keyword in their man pages

Tutorial Five

- File system security (access rights)
- Changing access rights
- Processes and Jobs
- Listing suspended and background processes
- Killing a process

Tutorial Five

`ls -lag`	list access rights for all files
`chmod [options] file`	change access rights for named file
`command &`	run command in background
`^C`	kill the job running in the foreground
`^Z`	suspend the job running in the foreground
`bg`	background the suspended job
`jobs`	list current jobs
`fg %1`	foreground job number 1
`kill %1`	kill job number 1
`ps`	list current processes
`kill 26152`	

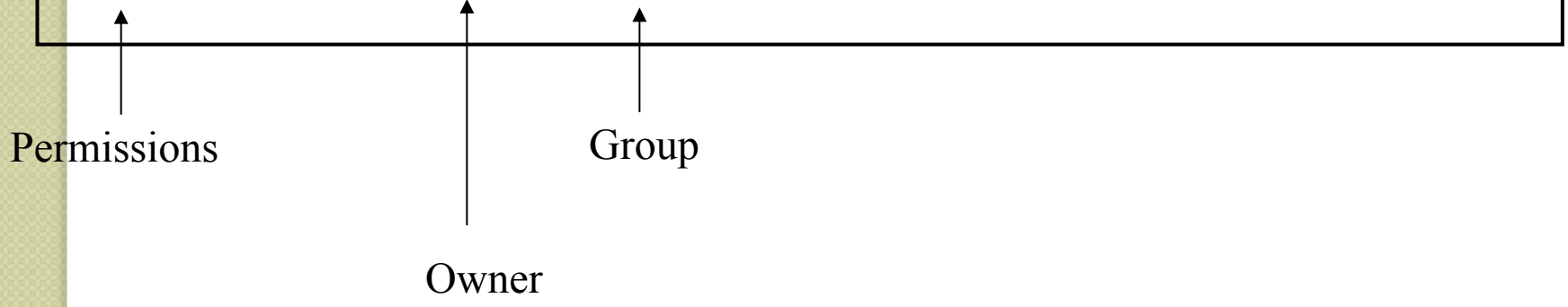
File Permissions

- Every file
  - Is owned by someone
  - Belongs to a group
  - Has certain access permissions for owner, group, and others

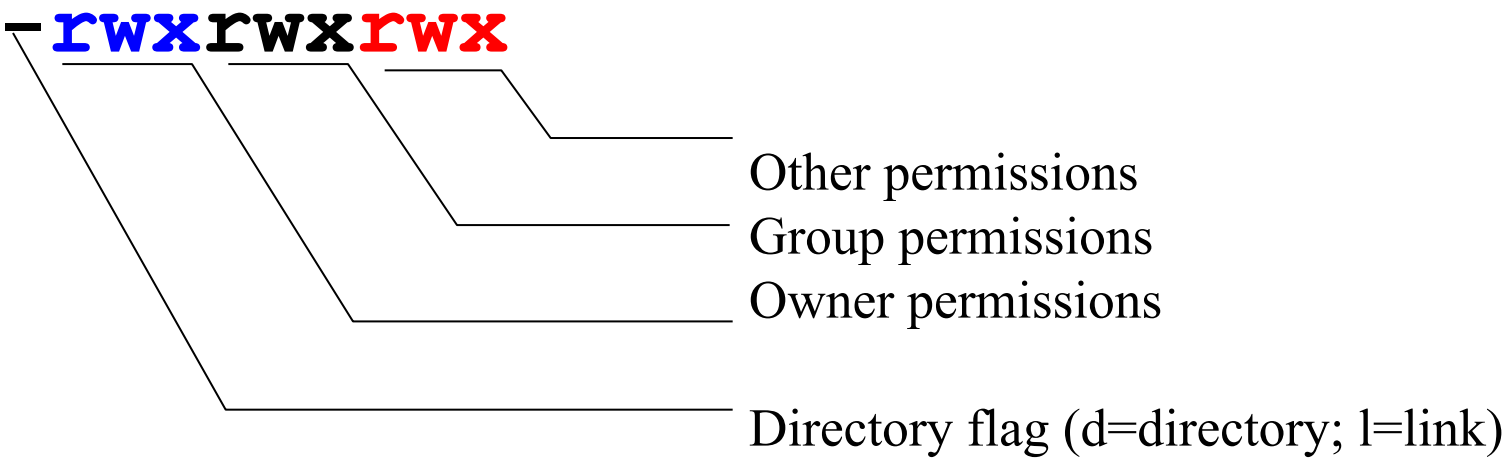
Check Permissions

- The long version of a file listing (`ls -l`) will display the file permissions:

```
-rwxrwxr-x   1 rvdheij   rvdheij      5224 Dec 30 03:22 hello
-rw-rw-r--   1 rvdheij   rvdheij       221 Dec 30 03:59 hello.c
-rw-rw-r--   1 rvdheij   rvdheij      1514 Dec 30 03:59 hello.s
drwxrwxr-x   7 rvdheij   rvdheij      1024 Dec 31 14:52 posixuft
```



Interpreting File Permissions



File Permissions

Permission type	When used with files	When used with directories
Read	Read a file or copy a file	List the contents of a directory
Write	Write to the file, including deleting the file	Create files
Execute	Execute programs and shell scripts, which are text files containing Linux commands	Modify the file permissions

File Permissions

- Permissions are set for user, group, and others
- Each permission is set with a single digit from 0 to 7 based on the combination of permissions
  - **read = 4=2^2 ---> 100**
  - **write = 2=2^1 ----> 010**
  - **execute = 1=2^0 ---- >001**
  - rwx=7=**1**x2^2+**1**x2^1+**1**x2^0-→**111**

Using chmod to Set Permissions

Command	Permissions		
	Owner	Group	Other
`chmod 755 myfile`	rwx	r-x	r-x
`chmod 540 myfile`	r-x	r--	---
`chmod 744 myfile`	rwx	r--	r--

Controlling Processes

- A process
  - Is a program that is currently executing
  - Can be created and destroyed
  - Has resources allocated to it
  - Has an environment associated with it that:
    - Process and process group IDs
    - Open files
    - Working directory
    - File creation mask
    - Real and effective user and group IDs
    - Resource limits: maximum file size, maximum amount of memory
    - Signal action settings
    - A set of named variables
  - Can create other processes
  - Can communicate with other processes

Controlling Processes

- Creating a process
  - Running jobs in the foreground: *command*
  - Running jobs in the background: *command &*
- Obtaining process status
  - jobs
    - Displays status of jobs in the current session
    - Job number, job status, PID
  - ps
    - Shows current status of processes
    - PID, state, accumulated execution time, command, ……
- Controlling and managing jobs
  - Placing a job in the foreground: *fg*
  - Restarting a job in the background: *bg*
  - Stopping a process: *Ctrl/C, kill*
  - Setting process priority: *nice*
  - Scheduling jobs to run at appropriate times: *at, crontab*

Tutorial Six

- *df*
- *du*
- *compress*
- *uncompress*
- *gzip*
- *file*
- *history*

More Commands

- **`tar`** - manipulates archives
  - An archive is a single file that contains the complete contents of a set of other files; an archive preserves the directory hierarchy that contained the original files.

```
tar -tzf imap-4.7.tar.gz
imap-4.7/
imap-4.7/src/
imap-4.7/src/c-client/
imap-4.7/src/c-client/env.h
imap-4.7/src/c-client/fs.h
```

Tutorial Seven

- Compiling UNIX software packages
- Download source code
- Extracting source code
- Configuring and creating the Makefile
- Building the package
- Running the software
- Stripping unnecessary code

Tutorial Eight

- *Setting Environment*
- *Setting paths*

Environment Variables

- Environment variables are global settings that control the function of the shell and other Linux programs. They are sometimes referred to global shell variables.
- Setting:
  - `VAR=/home/fred/doc`
  - `export TERM=ansi`
  - `SYSTEMNAME=\`uname -n\``

Environment Variables

- Using Environment Variables:
  - `echo $VAR`
  - `cd $VAR`
  - `cd $HOME`
  - `echo "You are running on $SYSTEMNAME"`
- Displaying - use the following commands:
  - **`set` (displays local & env. Vars)**
  - `export`
- Vars can be retrieved by a script or a program

Some Important Environment Variables

- HOME
  - Your home directory (often be abbreviated as "~")
- TERM
  - The type of terminal you are running (for example vt100, xterm, and ansi)
- PWD
  - Current working directory
- PATH
  - List of directories to search for commands

END