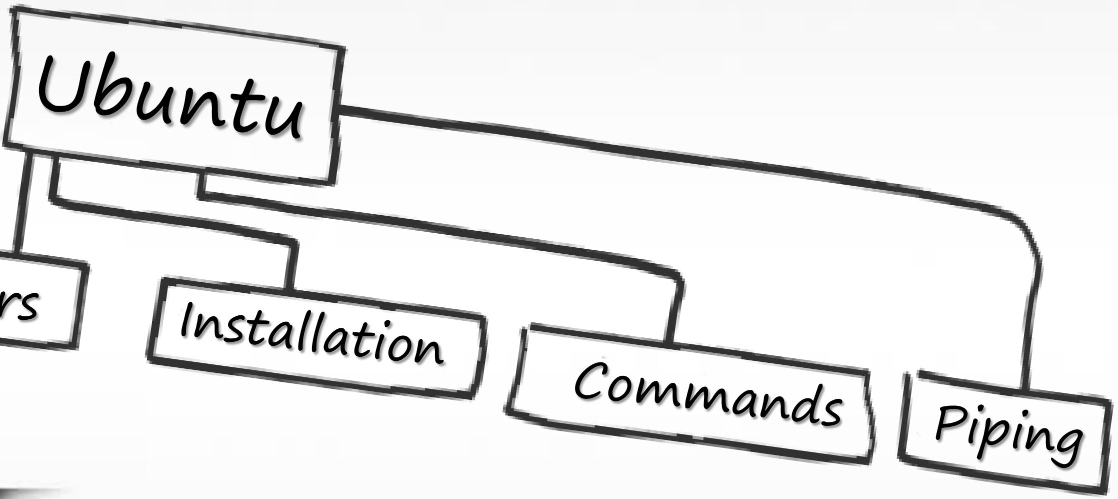
Ubuntu Fundamentals

NOW



Course Materials

You can access the course materials via this link

http://goo.gl/MZqU4b

Day 1 Contents

- Free/Open Source Software and Licenses.
- Linux History.
- Linux Components.
- Installation
- Basic Commands
- Linux Documentation
- File and Directory Basics

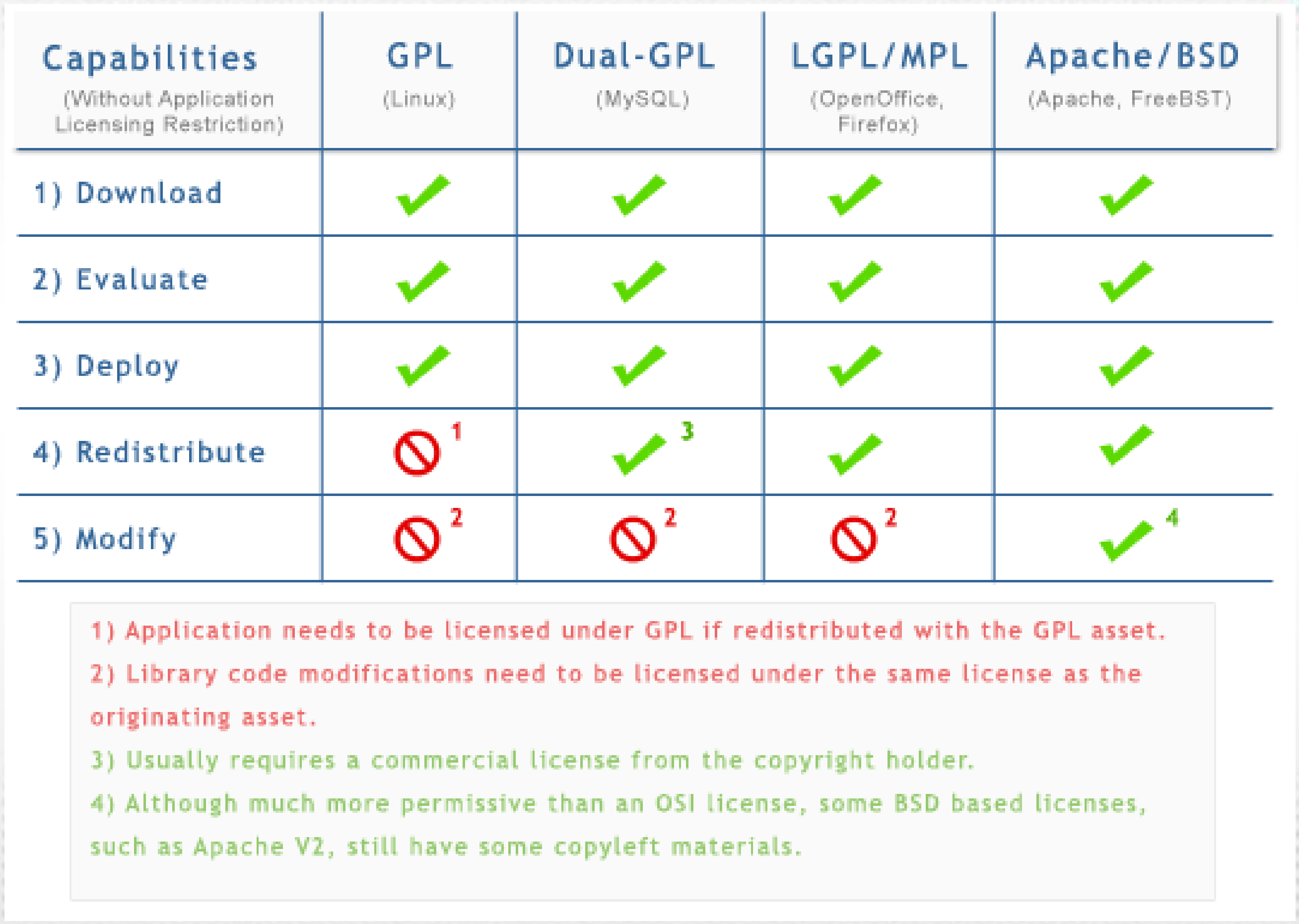
What is FOSS?

- Free/Open Source Software (FOSS) provides many freedoms, including the ability to:
  - View the source code used to compile programs
  - Make modifications
  - Distribute these modifications.
- Where is the benefit ?
  - Customers are usually willing to pay for training, support and consultation
- Most FOSS is covered under a public license. The most common public license is the GNU General Public License (GPL).

FOSS Licenses

- An open-source license is a type of license for computer software and other products that allows the source code, blueprint or design to be used, modified and/or shared under defined terms and conditions.
- Examples:
  - GPL, LGPL, Apache, Mozilla Public License and BSD.

FOSS Licenses Comparison



Capabilities (Without Application Licensing Restriction)	GPL (Linux)	Dual-GPL (MySQL)	LGPL/MPL (OpenOffice, Firefox)	Apache/BSD (Apache, FreeBST)
1) Download	✓	✓	✓	✓
2) Evaluate	✓	✓	✓	✓
3) Deploy	✓	✓	✓	✓
4) Redistribute	🚫 [1]	✓ [3]	✓	✓
5) Modify	🚫 [2]	🚫 [2]	🚫 [2]	✓ [4]

1) Application needs to be licensed under GPL if redistributed with the GPL asset.

2) Library code modifications need to be licensed under the same license as the originating asset.

3) Usually requires a commercial license from the copyright holder.

4) Although much more permissive than an OSI license, some BSD based licenses, such as Apache V2, still have some copyleft materials.

Linux History

- Unix first version created in Bell Labs in 1969
- Unix flavors
  - IBM->AIX, Hewlett-Packard->HP/UX, Sun-> Solaris and Silicon Graphics->IRIX
- Operate in a same manner
- Offer the same standard utilities and commands

- Linus Torvalds
- Finished his college in 1991
- Created Linux kernel

Linux History (cont'd)

- Linux Flavors
  - Debian GNU/Linux
  - Gentoo Linux
  - Mandriva
  - MkLinux
  - Red Hat Enterprise Linux
  - Rock Linux
  - Slackware Liunx
  - SUSE Linux
  - Yellow Dog Linux
  - TurboLinux
  - ASPLinux
  - ScrudgeWare
  - Xandros
  - KNOPPIX
  - Fedora
  - Symphony OS
  - Ubuntu Linux

Ubuntu History

- Ubuntu based on Debian GNU/Linux distribution and distributed as free and open source software.
- It is named after the Southern African philosophy of Ubuntu ("humanity towards others").
- Ubuntu is designed primarily for desktop usage, Web statistics suggest that Ubuntu's share of Linux desktop usage is about 50 percent, and upward trending usage as a web server.

Ubuntu Releases

- The Ubuntu team broke new ground in committing to a program of scheduled releases on a predictable **six-month** basis. It was decided that every fourth release, issued on a **two-year basis**, would receive **long-term support (LTS)**. LTS releases are typically used for large-scale deployments.

Why Linux?

- Linux is growing in the home users sector and the dominant of the professional and servers sector.
- Internet service providers (ISPs), e-commerce sites, and other commercial applications all use Linux today and continue to increase their commitment to Linux.

Installation

- Ubuntu Desktop Edition
  - 700 MHz processor (about Intel Celeron or better)
  - 512 MiB RAM (system memory)
  - 5 GB of hard-drive space (or USB stick, memory card or external drive but see LiveCD for an alternative approach)
  - VGA capable of 1024x768 screen resolution
  - Either a CD/DVD drive or a USB port for the installer media

Installation

- Ubuntu Server (CLI) Installation
  - 300 MHz x86 processor
  - 192 MiB of system memory (RAM)
  - 1 GB of disk space
  - Graphics card and monitor capable of 640x480
  - CD drive

Types of Installation

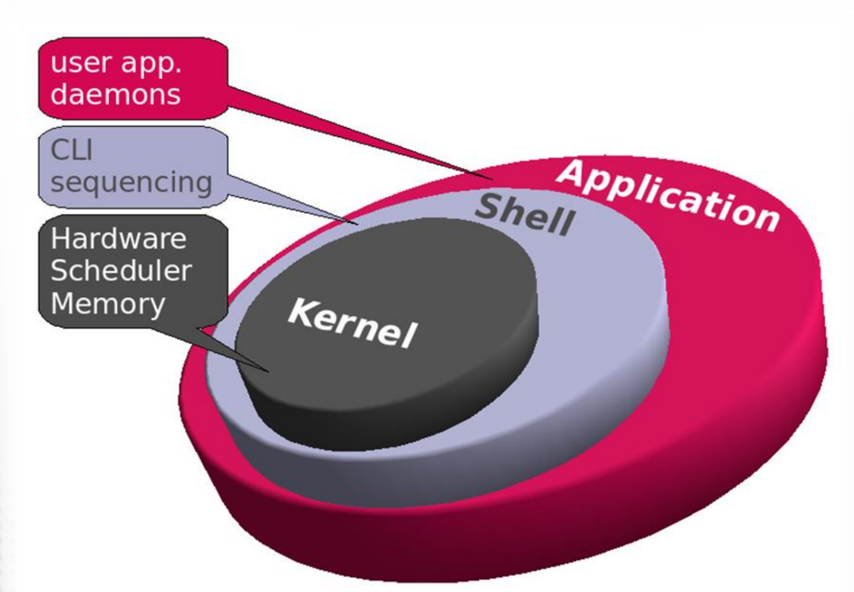
- Kickstart Mode
  - Permits automated installation
- Graphical Installation
- Text Based Installation

Linux Components

- **Kernel**
  - Is the core of the operating system.
  - Contains components like device drivers.
  - It loads into RAM when the machine boots and stays resident in RAM until the machine powers off.
- **Shell**
  - Provides an interface by which the user can communicate with the kernel.
  - “bash” is the most commonly used shell on Linux.
  - The shell parses commands entered by the user and translates them into logical segments to be executed by the kernel or other utilities.

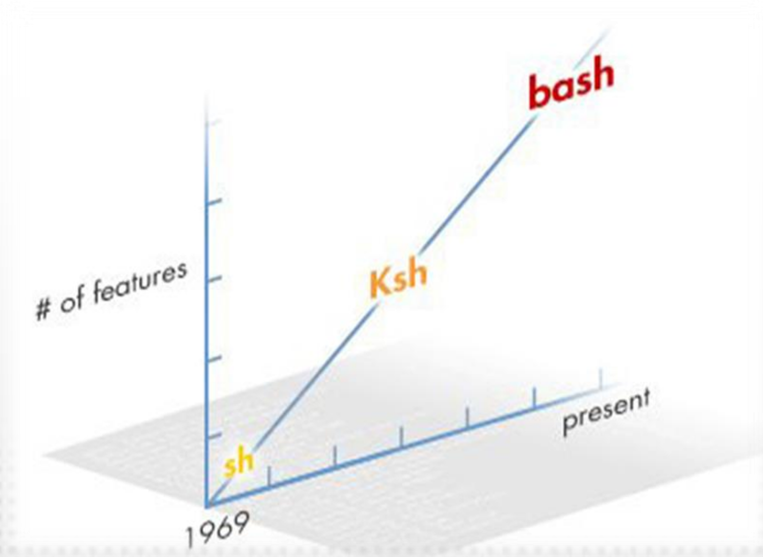
Linux Components

- **Terminal**
  - Gives the shell a place to accept typed commands and to display their results

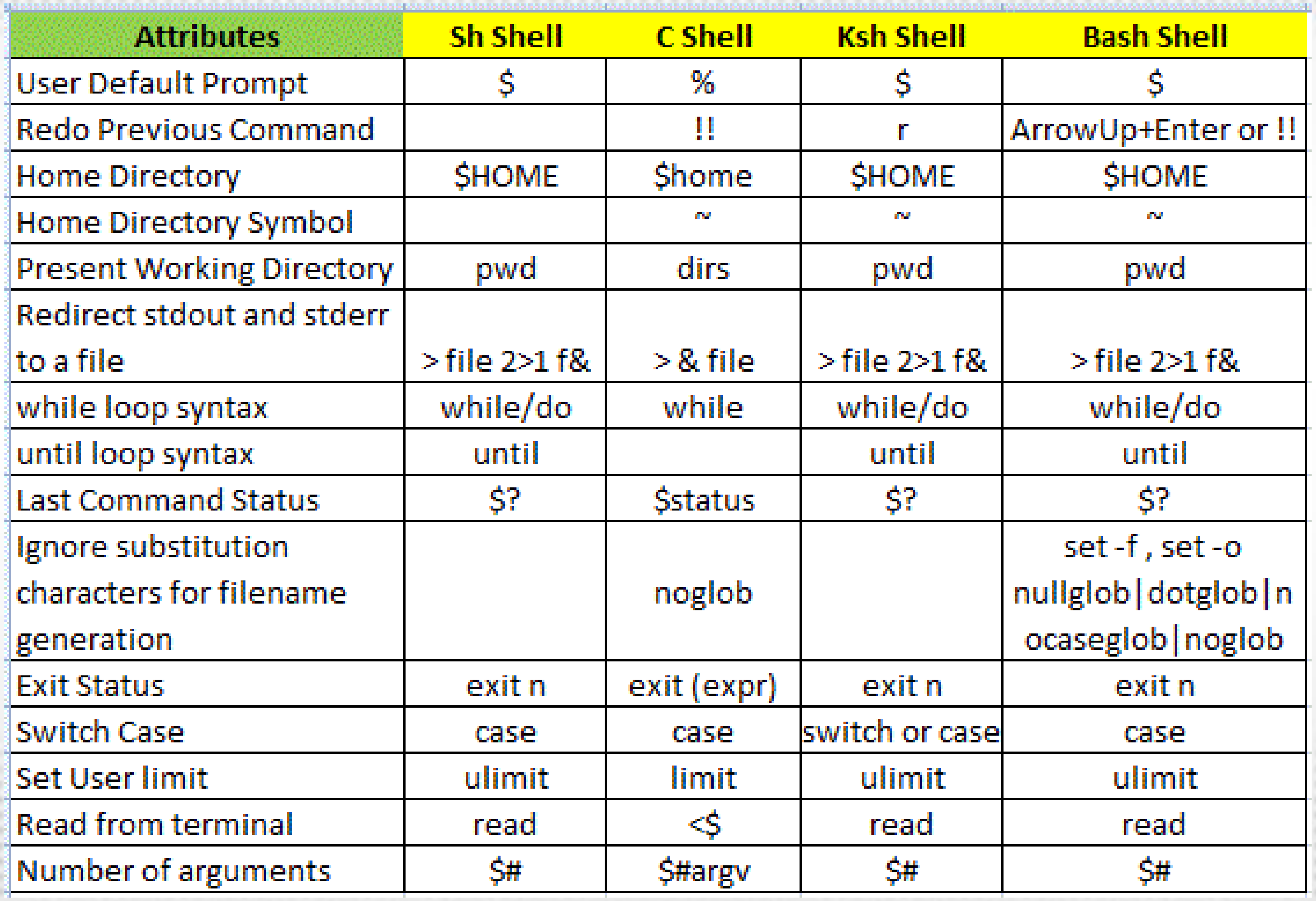


Command-Line Shells

- There are lot of shells as : Bourn Shell (sh), Korn Shell (ksh), C Shell (csh) and Bourn Again Shell (bash). They have different features that will be discussed later.



Command-Line Shells



Attributes	Sh Shell	C Shell	Ksh Shell	Bash Shell
User Default Prompt	$	%	$	$
Redo Previous Command		!!	r	ArrowUp+Enter or !!
Home Directory	$HOME	$home	$HOME	$HOME
Home Directory Symbol		~	~	~
Present Working Directory	pwd	dirs	pwd	pwd
Redirect stdout and stderr to a file	> file 2>1 f&	> & file	> file 2>1 f&	> file 2>1 f&
while loop syntax	while/do	while	while/do	while/do
until loop syntax	until		until	until
Last Command Status	$?	$status	$?	$?
Ignore substitution characters for filename generation		noglob		set -f , set -o nullglob|dotglob|nocaseglob|noglob
Exit Status	exit n	exit (expr)	exit n	exit n
Switch Case	case	case	switch or case	case
Set User limit	ulimit	limit	ulimit	ulimit
Read from terminal	read	<$	read	read
Number of arguments	$#	$#argv	$#	$#

Running commands

- Commands have the following syntax:

```
command [options] [arguments]
```

- Each item is separated by a space.
- Options modify the command's behavior.
- Arguments are files name or other information needed by the command.
- Separate commands with semicolon (;).

Examples

```
uname
```

Linux

```
uname -n
```

host1

```
uname -a
```

Linux host1 ……………….

Examples

```
cal
September 2010
 S  M Tu  W Th  F  S
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
```

Examples

```
cal 5 2004
```

```
May 2004
 S  M Tu  W Th  F  S
                   1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30 31
```

```
cal ;uname
cal 5 2002; date; uname
```

Interrupting command execution

- To interrupt a command that's taking too long to execute, use **[Ctrl]-c**.
- Occasionally, you might enter a command without an argument that expects input to come from the keyboard. In this case, use **[Ctrl]-d** to terminate the command.

Linux Documentation

Manual page consists of:

- Name
  - The name of the command and a one-line description
- Synopsis
  - The syntax of the command
- Description
  - Explanation how the command works and what it does
- Files
  - The file used by the command
- Bugs
  - Known bugs and errors
- See also
  - Other commands related to this one

Linux Documentation

man –k keyword

Shows the commands that have manual pages that contains any of the given keywords.

man –s keyword

whatis command

Shows the commands one line description

Linux Documentation

--help Option

- Another way to get help about a command.
- help is built in the command itself (if supported).

Linux Documentation

HOWTO Documents

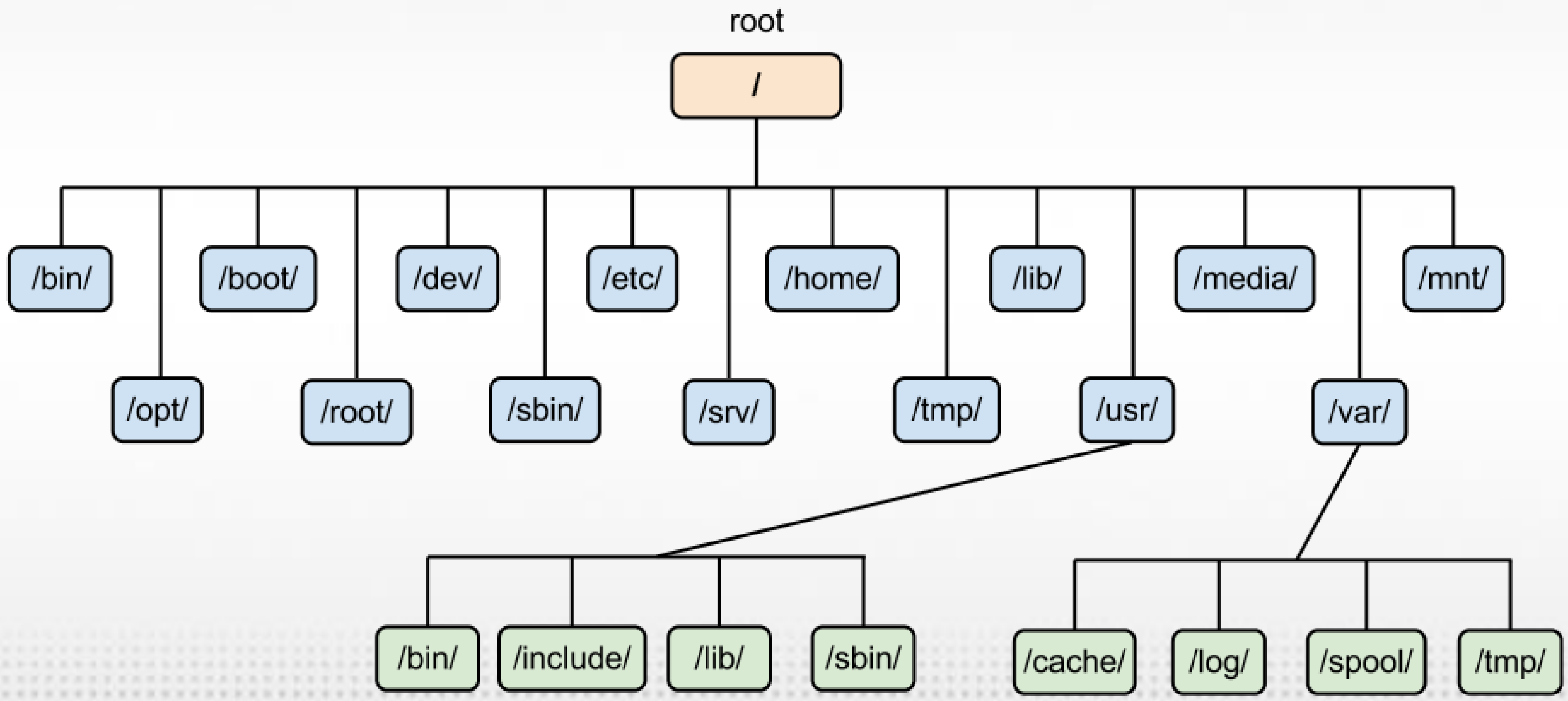
- Documents which describe in details a certain aspect of configuring or using Linux.
- They are text files in /usr/share/doc/HOWTO
- Need to be installed manually

Introduction to Directories

- Think of
  - File system as a building
  - Directory is a room
  - File is a desk
- The current working directory is the room you are.
- To find out where you are at any time

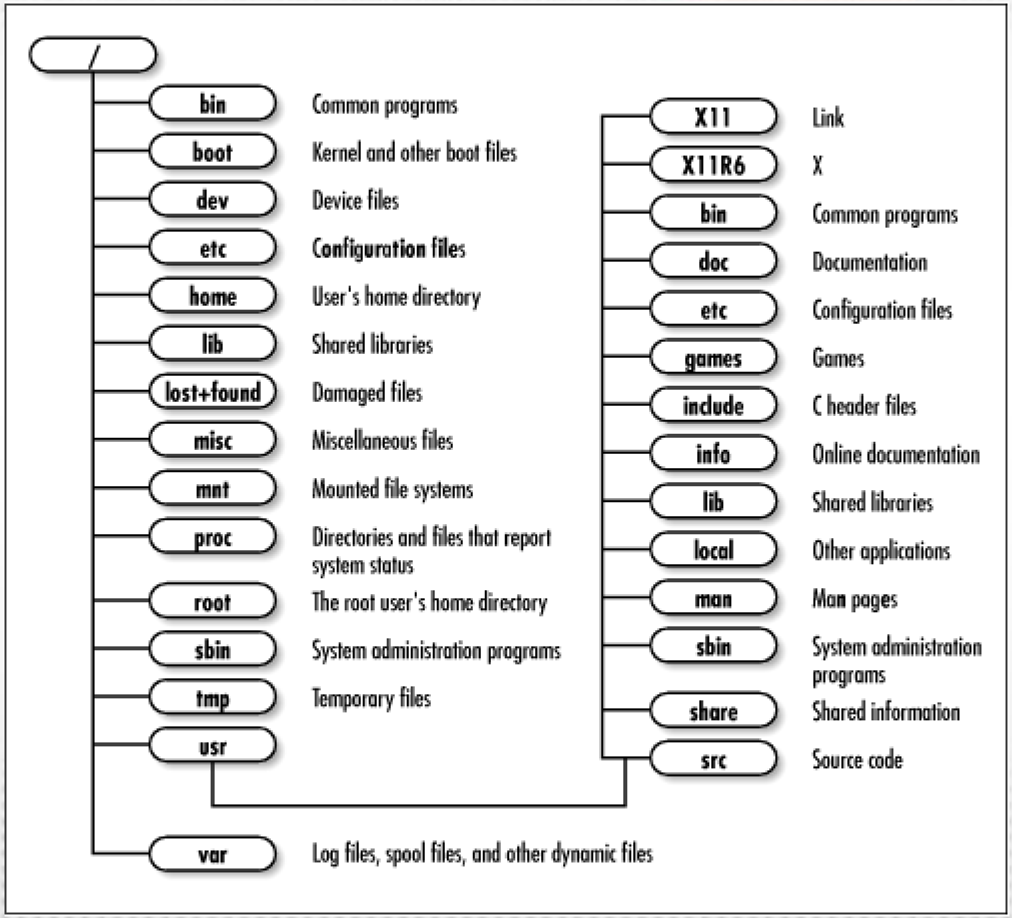
```
pwd
/home/guest
```

How Directories Work ?



How Directories Work ?

- Pathnames
  - Absolute pathname
  - Relative pathname



Changing Directories

- To move from directory to directory on the system

```
cd /home/user1/work
cd ..
cd ~
cd -
```

Listing Directory Contents

```
ls
dir1   dir2        file1
dir3   file2            file3

ls /home/user1/dir1
f1        f2

pwd
/home/user1

ls dir1
f1        f2
```

Listing Directory Contents

```
ls -a dir1
.           .f1   f1
..          .f2  f2

ls -l dir1
total 2
-rw-r--r-- 1 islam islam 20 2 May 21 16:11 f1
-rw-r--r-- 1 islam islam 20 0 May 21 16:11 f2

ls -F
dir1/ dir2/       file1
dir3/ file2*           file3@
```

Listing Directory Contents

```
ls -ld dir1
drwxr-xr-x     2 sbahader ssdp20 512 May 21
16:06 dir1


ls -R
.:
dir1   dir2        file1
dir3   file2       file3
./dir1:
f1        f2
./dir2:
./dir3:
```

Checking Free Space

- The `df` command displays number of free disk blocks and files.

```
df [-h] [block_device| directory|file]
```

- Example

```
df -h /
Filesystem size used avail capacity Mounted on
/dev/hda0  15G  976M  14G   6%         /
```

Checking Free Space

- The `du` command display the total sum of space allocated to all files hierarchy rooted in the directory specified.

```
du [-sh] [dir…]
```

- Example

```
du -sh
  14K
```

File Naming

- File names may be up to 255 characters.
- There are no extensions in Linux
- Avoid special characters as >< ? * # '
- File names are case sensitive

Viewing File Content

```
cat fname
more fname
```

- Scrolling keys for the more command
  - Spacebar: moves forward on screen
  - Return: scroll one line at a time
  - b: move back one screen
  - /string: search forward for pattern
  - n: find the next occurrence
  - q: quit and return to the shell prompt

```
head -n fname
tail [-n|+n] fname
```

File Globing

- When typing commands, it is often necessary to issue the same command on more than one file at a time.
- The use of wildcards, or “metacharacters”, allows one pattern to expand to multiple filenames

Metacharacters

- Asterisk(*): represents 0 or more character, except leading (.)

Example:

```
ls f*
file.1 file.2 file.3 file4
file1 file2 file3 fruit

ls *3
file.3 file3
dir3:
moon planets space sun
```

Metacharacters

- Question mark(?) character represents any single character except the leading (.)

Examples

```
ls file?
file4 file1 file2


ls z?
z?: No such file or directory
```

- Square bracket([]): represent a range of characters for a single character position.

```
Example
ls [a-f]*
ls [pf]*
```

Metacharacters

```
 ls -a
. .. .profile abm bam bat battle project
ls -l b*
-rw-r----- 1 sgs 16 Feb 12 11:04 bam
-rw-r----- 1 sgs 12 Feb 12 11:05 bat
-rw-r----- 1 sgs 19 Feb 12 11:06 battle
ls *
abm bam bat battle project
ls  .*
. .. .profile
ls *m
abm bam
ls *a*
abm bam bat battle
```

Metacharacters

```
ls   ???
abm bam bat
ls   ?a?
bam bat
ls   ?a*
bam bat battle
ls [ab]*
abm bam bat battle
ls -l [ab]m
ls: "[ab]m: No such file or directory
ls   [a-zA-Z]*
abm bam bat battle project
```

File and Directory Manipulation

- Coping Files and Directories

```
cp options source(s) target
```

Option	Description
-i	Prevents you from accidentally overwriting existing files or directories
-r	Copy a directory including the contents of all subdirectories

File and Directory Manipulation

- Moving and Renaming Files and Directories

```
mv options source(s) target
```

Option	Description
-i	Prevents you from accidentally overwriting existing files or directories

File and Directory Manipulation

- To create files

```
touch file(s)_name
```

- To create directories

```
mkdir [-p] dir(s)_name
```

File and Directory Manipulation

- To remove files

```
rm [-i] file(s)_name
```

- To remove directories

```
rmdir dir(s)_name
rm [-r] dir(s)_name
```