Overview

- Introduction – Linux philosophy
- Command line basics – getting help
- The shell revisited: some features
  - Navigating the file system
  - File manipulation
- Text editing
- Various commands
- Archiving
- Groups, users, security
- Process control

1

Linux filesystem
Navigating the filesystem

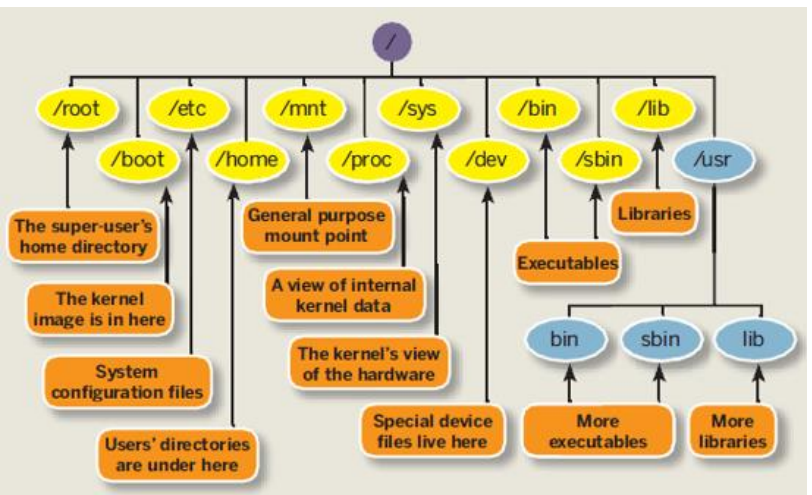
2

Linux File System

- *hierarchical directory structure*: files are organized in a tree-like pattern of directories (folders), which may contain files and other directories, etc.
- Everything is a file:
  - Regular files
  - Directories: files listing a set of files
  - Symbolic links: files referring to the name of another file
- *root /: th*e first directory in the file system.
- Note: comparison with Windows,
  - Windows has a separate file system tree for each storage device (e.g. C-drive, D-drive, I-drive, …)
  - Linux has a single file system tree, regardless of how many drives or storage devices are attached to the computer.
    Storage devices are attached (or *mounted*) at various points on the tree.

3

Linux File System



Source: http://linuxsuperuser07.blogspot.be/2011/09/rhel-6-file-system.html

4

Linux File System

Not imposed by the system. Can vary from one system to the other, even between two GNU/Linux installations!

`/`	Root directory
`/bin/`	Basic, essential system commands
`/boot/`	Kernel images, initrd, configuration files
`/dev/`	Files representing devices
`/etc/`	System configuration files
`/home/`	User directories
`/lib/`	Basic system shared libraries
`/media/`	Mount points for removable media

5

Linux File System

`/lost+found/`	Corrupt files the system tried to recover
`/mnt/`	Mount points for temporarily mounted filesystems
`/opt/`	Specific tools installed by the sysadmin
`/usr/local/`	often used instead
`/proc/`	Access to system information
`/sbin/`	Administrator-only commands
`/sys/`	System and device controls
`/tmp/`	Temporary files

The Unix filesystem structure is defined by the Filesystem Hierarchy Standard (FHS):
http://www.pathname.com/fhs/pub/fhs-2.3.html

6

Linux File System

- A **file** is a collection of data, with a location in the file system called a **path**. Paths will typically be a series of words (directory names) separated by forward slashes, /. Files are generally created by users via text editors, compilers, or other means.
- A **directory** is a special type of file. Linux uses a directory to hold information about other files, the equivalent of a folder in Windows. You can think of a directory as a container that holds other files or directories.
- A file is typically stored on physical storage media such as a disk (hard drive, flash disk, etc.).
- Every file must have a name because the operating system identifies files by their name.
  - File names may contain any characters, although some special characters (such as spaces, quotes, and parenthesis) can make it difficult to access the file, so you should avoid them in filenames.
  - File names can be as long as 255 characters, so use descriptive names.
  - File names are case sensitive,
  - A hidden file is any file that begins with a "." (not seen with the bare `ls`)

https://cvw.cac.cornell.edu/Linux/files

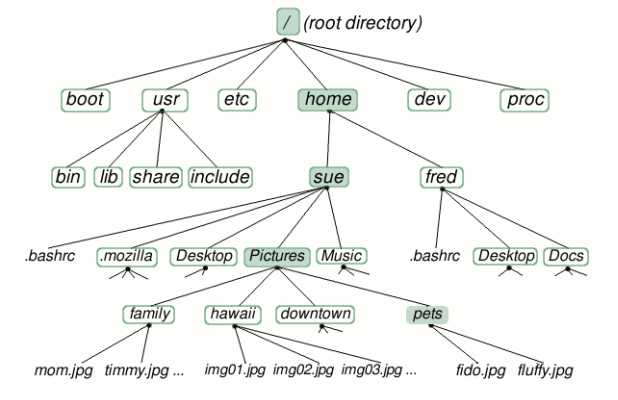
7

Linux File System

- "\" vs. "/":
  - In Linux, the "/" is the directory separator, and the "\" is an escape character.
  - In Windows, the forward-slash "/" is the command argument delimiter, while the backslash "\" is a directory separator
- Filenames:
  - In Linux, there is no such thing as a file extension.
    Periods can be placed at any part of the filename, and "extensions" may be interpreted differently by all programs, or not at all.
  - Windows uses the ".extension" filename convention, (e.g. FILENAME.TXT).

8

Linux File System-home directory



Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

9

ls command

Lists the files in the current directory, in alphanumeric order, except files starting with the "." character.

- `$ ls -a` (all)
  Lists all the files (including .* files)
- `$ ls -l` (long)
  Long listing (type, date, size, owner, permissions)
- `$ ls -t` (time)
  Lists the most recent files first
- `$ ls -S` (size)
  Lists the biggest files first
- `$ ls -r` (reverse)
  Reverses the sort order
- `$ ls -ltr` (options can be combined)
  Long listing, most recent files at the end

10

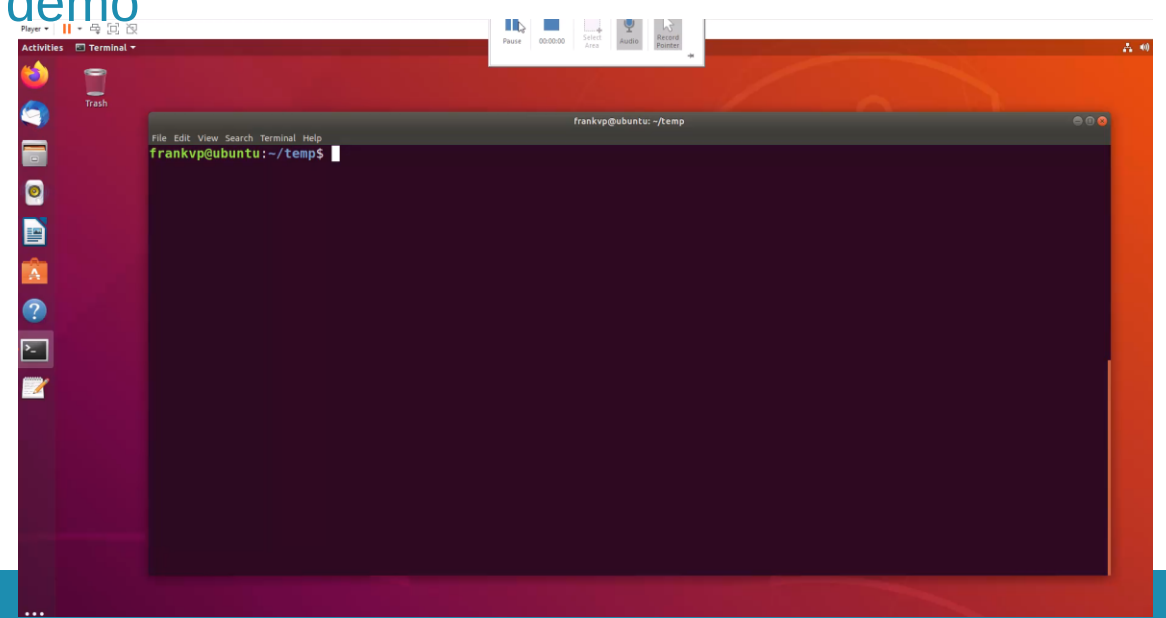
ls command

- `$ ls *txt`
  The shell first replaces *txt by all the file and directory names ending by txt (including .txt), except those starting with ., and then executes the ls command line.
- `$ ls -d .*`
  Lists all the files and directories starting with .
  -d list directories themselves, not their contents.
  `$ ls -d */`
- `$ ls ?.log`
  Lists all the files which names start by 1 character and end by .log

https://www.thegeekstuff.com/2009/07/linux-ls-command-examples/

11

demo



12

Moving around

- Display the current/working directory
  - `$ pwd`
  - **P**rint **W**orking **D**irectory
  - displays your current location within the file system.
- Change (navigate) directories.
  - `$ cd dir_name`
  - **C**hange **D**irectories
  - changes the position to the specific directory
- You can specify directory names in two ways:
  - Absolute pathname (starts from the root of the tree)
    ```
    $ cd /u/home/hpc/test/bin
    ```
  - Relative pathname (relative to your current directory)
    ```
    $ cd
    $ cd .
    $ cd ..
    $ cd test/bin
    ```

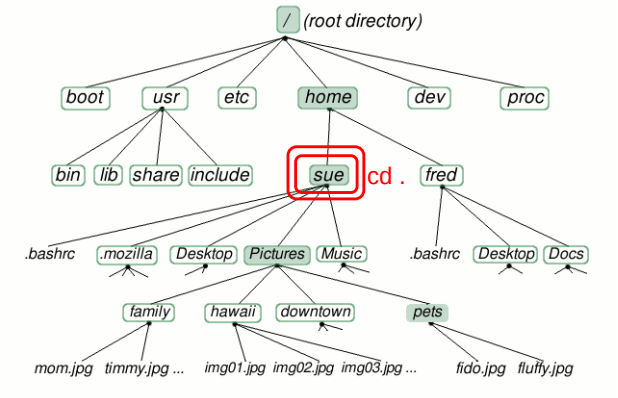
13

Special directories

- a few characters representing shortcuts to locations.
- **(single dot) .**
  - The current working directory.
  - Useful to run commands in the current directory
  - ./readme.txt and readme.txt are equivalent.
- **(double dot) ..**
  - The parent (enclosing) directory. Always belongs to the . Directory
  - Typical usage:
    `cd ..`
- **(tilde) ~**
  - Shells just substitute it by the home directory of the current user.
- **(dash) -**
  - `cd –` jump back to the previous directory

14

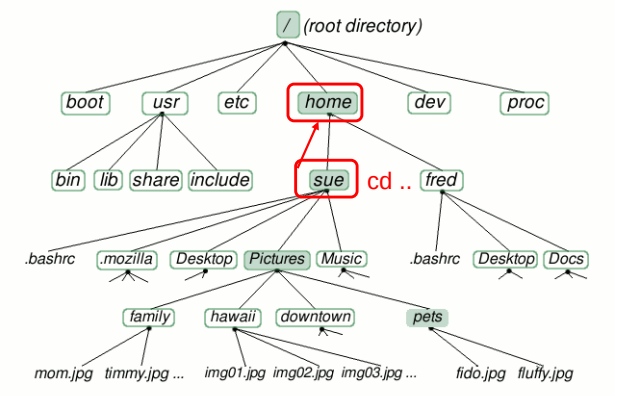
Linux File System - directories



Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

15

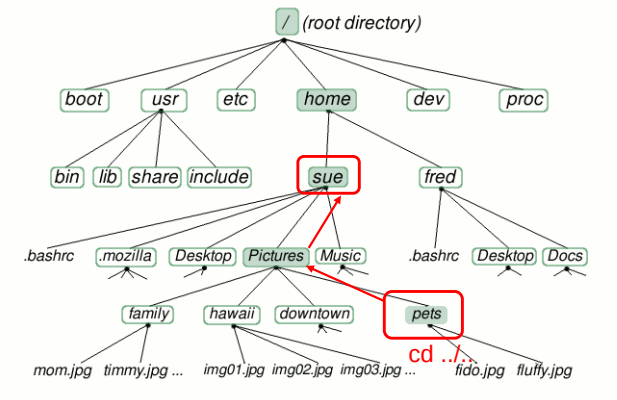
Linux File System - directories



Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

16

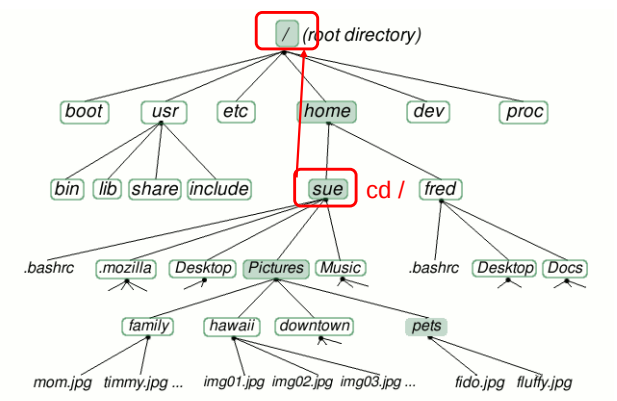
Linux File System - directories



Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

17

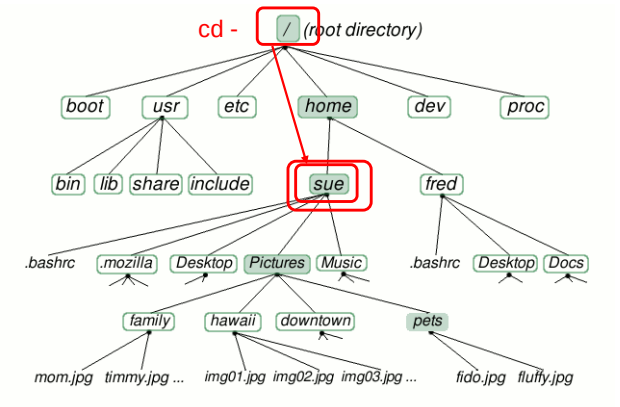
Linux File System - directories



Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

18

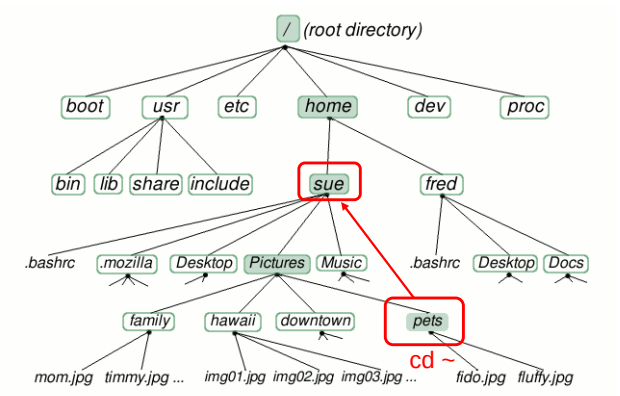
Linux File System - directories



Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

19

Linux File System - directories

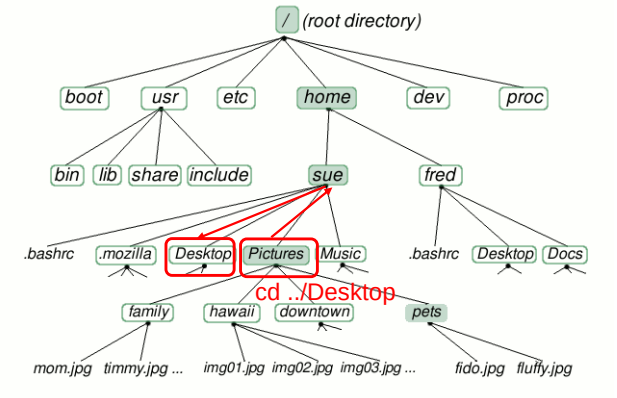


Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

20

Linux File System-home directory

Relative path

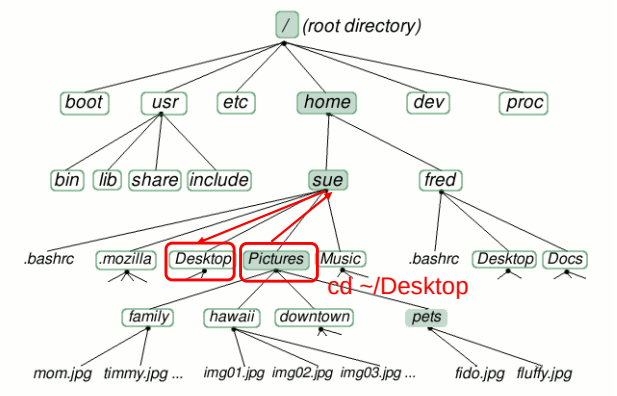


Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

21

Linux File System - directories

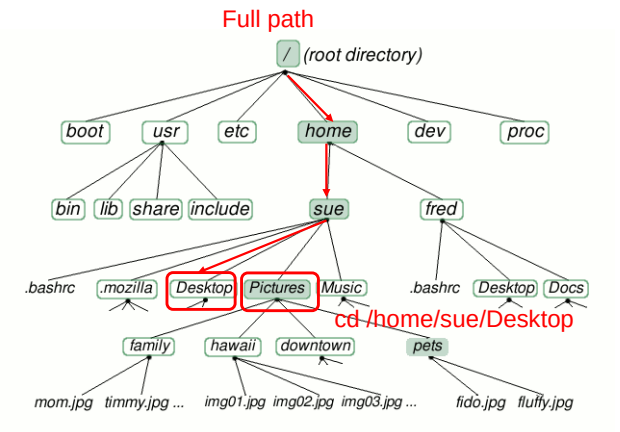
Relative path



Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

22

Linux File System - directories



Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

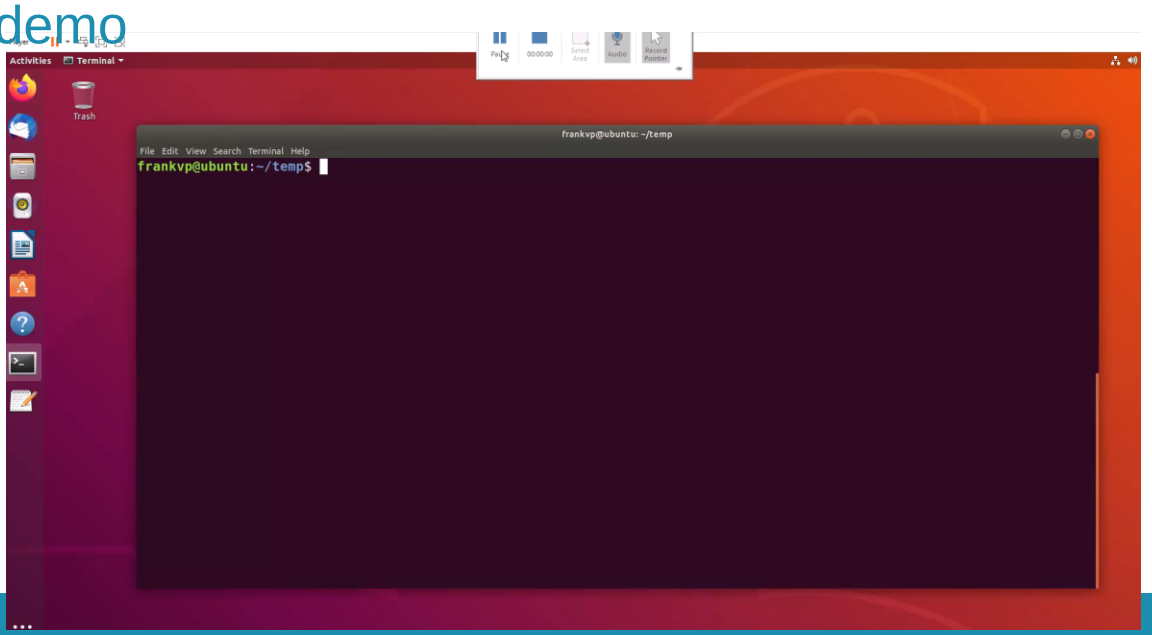
23

File paths

- A path is a sequence of nested directories with a file or directory at the end, separated by the / character
- **Relative path**: `documents/fun/file1`
  Relative to the current directory
- **Absolute path**: `/home/user/leuven/file2`
- `/` : root directory.
  Start of absolute paths for all files on the system

24

demo



25

File manipulation

26

File Manipulation

- For all file manipulation commands relative of absolute paths can be used (or just files in the current directory when no extra path specified)
- Most of the commands are intuitive – shortcuts of English names

27

Directories

- Create directories
  - The `mkdir` command is used to create directories
  - `$ mkdir dir1 dir2 dir3`
- Remove directories
  - The `rmdir` command removes directories
  - `$ rmdir dir1`
  - `rmdir` will only remove empty directories.
    To remove a non-empty directory, use
    `$ rm -r [DIRECTORY]` instead. (BE CAREFUL!)

28

Copy a file

- The `cp` command copies files and directories
- The default behavior will overwrite any existing file(s). The **-i** option overrides this behavior and prompts the user before overwriting the destination file.
- **syntax: `cp [OPTIONS] [SOURCE] [DESTINATION]`**
  - `$ cp <source_file> <target_file>`
    Copies the source file to the target.
  - `$ cp file1 file2 file3 ... dir`
    Copies the files to the target directory (last argument).
  - `$ cp -i` (interactive)
    Asks for user confirmation if the target file already exists
  - `$ cp -r <source_dir> <target_dir>` (recursive)
    Copies the whole directory.
  - `$ cp -v` (verbose)
    Displays what has been copied

29

Move or rename files

- Move or rename files and directories: `mv`
- The default behavior will overwrite any existing file(s).
- **syntax: `mv [OPTIONS] [SOURCE] [DESTINATION]`**
  - `$ mv old_name new_name`
    Renames the given file or directory.
  - `$ mv -i` (interactive)
    If the new file already exits, asks for user confirm
- The `mv` command can also be used to move or rename directories
  - `$ mv NewFiles/ OldFiles/`
  - -r option is not necessary

30

Remove files

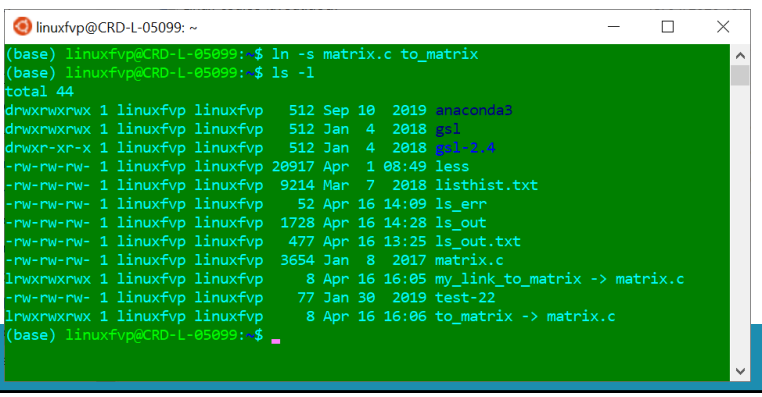
- The rm command removes files.
- `$ rm file1 file2 file3 ...`
  Removes the given files.
- `$ rm -i` (interactive)
  Always ask for user confirmation.
- `$ rm -r dir1 dir2 dir3` (recursive)
  Removes the given directories with all their contents.

- Tip:
  Whenever you use wildcards with `rm` (besides carefully checking your typing!), test the wildcard first with `ls`. This will let you see the files that will be deleted. Then press the up arrow key to recall the command and replace the `ls` with `rm`.

31

Create links

- **Soft link**: similar to a shortcut in Windows. It is an indirect pointer to a file or directory; can point to a file or a directory on a different filesystem or partition.
- Symbolic links are created when using the `-s` option with the `ln` command.
  `ln -s [OPTIONS] FILE LINK`
- Check with `ls -l`
- The first character "l", indicates that the file is a symlink.
- The "->" symbol shows the file the symlink points to.



33

Create links

- Editing a symbolic link file is the same as editing the source file
- Deleting the symbolic link does not delete the source file.
- Deleting the source file leaves a dangling link
- `$ ln -s file_v5.doc file_final.doc`
  creates a symbolic link called file_final.doc that points to file_v5.doc
- `$ ln -s /home/demo/dir1/dir2/dir3 /home/demo/jump2dir`
  creates a symbolic link called jump2dir that points to a deep directory - allows for quicker access)

34

More on files

- Search for files and directories
  - The **`find`** command performs a raw search on a file system to locate the specified items.
  - `$ find  location –name some-name`
    `($ find / -name matrix.c)`
  - You can also specify more than one location to search,
- Search the locate database for files and directories
  - The **`locate`** command displays the location of files that match the specified name.
  - Faster than find but lacks the ability to search for advanced characteristics such as file owner, size, and modification time.

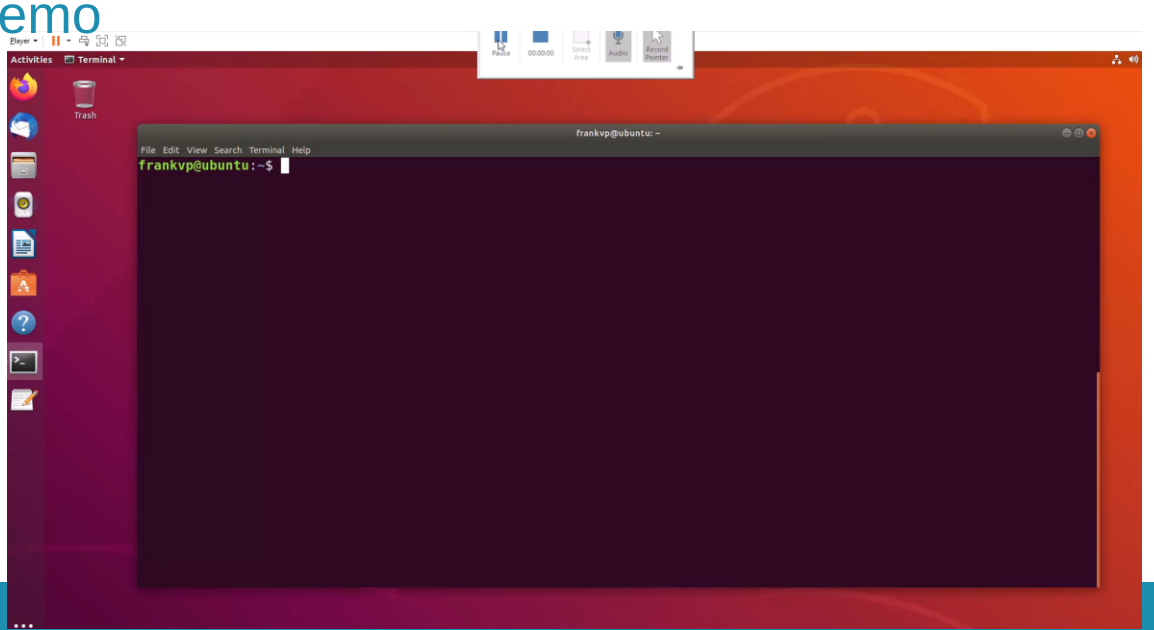
36

More on files

- Display extended information about a file system, file, or directory
- What does a file contain?
  - Determine a file's type: **`file`**
  - will print a brief description of the file's contents
  - `$ file  filename`
- The `stat` command displays extended information about files. It includes helpful information not available when using the `ls` command

37

demo



38

Hands-on 3

39