Outline

- Introduction - history
- Command line basics – getting help
- ➢File system
- Working with files and directories
- More file handling
- The shell revisited
- Monitoring resources

1

Linux filesystem

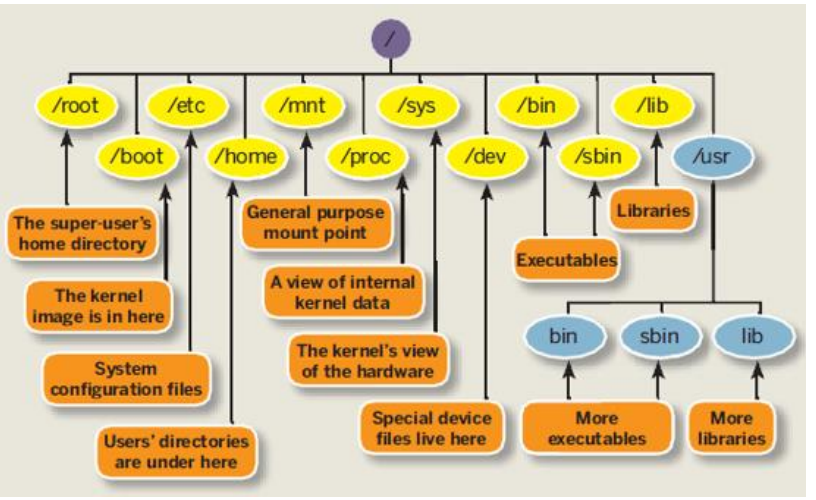
2

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 /: the* 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):
https://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

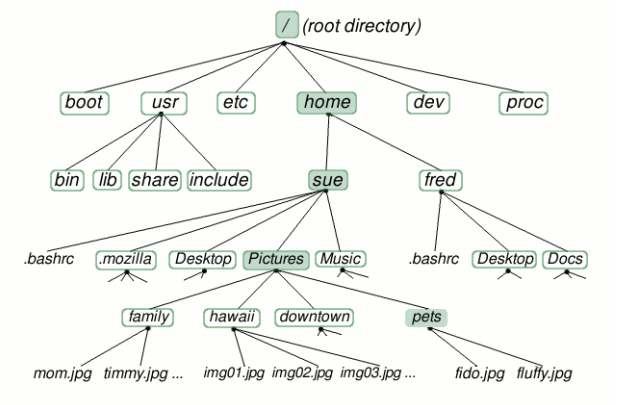
7

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

Useful command

- `tree`: recursively list or display the content of a directory in a tree-like format.
  - directory paths and files in each sub-directory
  - summary of a total number of sub-directories and files.
- `tree -L 2` limit the depth
- See also: https://www.tecmint.com/linux-tree-command-examples/

10

Navigating the filesystem: ls

11

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

12

ls command

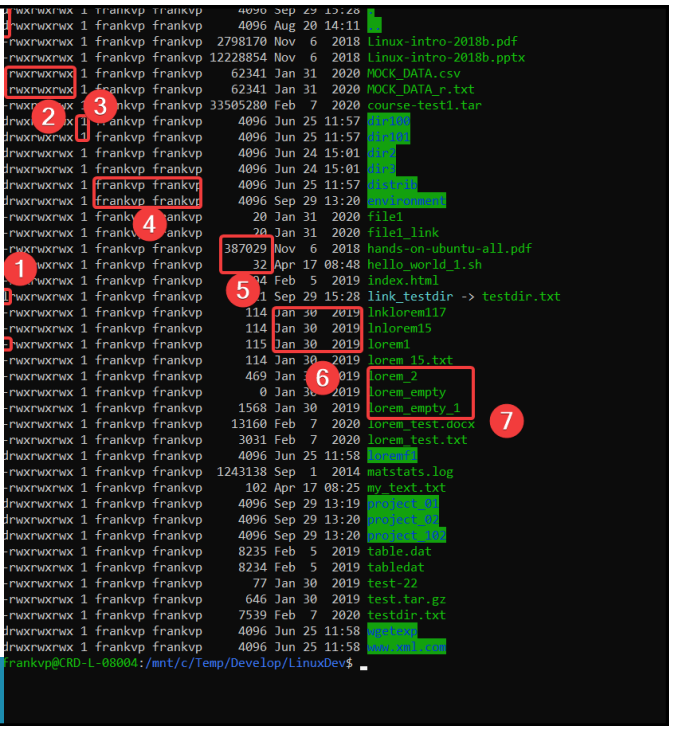
- `$ ls *txt`
  The shell first replaces *txt by all the file and directory names ending by txt (including .txt), except those starting with ., and then executes the ls command line.
- `$ ls -F`
  Display file type
  / directory
  * executable
- `$ ls ?.log`
  Lists all the files which names start by 1 character and end by .log

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

13

ls -l

1. Type
   - file
   d directory
   l link
2. Permission (Read, Write, eXecute) for Owner Group World
3. Links (number)
4. Owner (+ group)
5. Size
6. Last modification date
7. Name



14

Globbing: use wildcards

Wildcard	Function
*	Matches 0 or more characters
?	Matches 1 character
[abc]	Matches one of the characters listed
[a-c]	Matches one character in the range
[!abc]	Matches any character not listed
[!a-c]	Matches any character not listed in the range
{tacos,nachos}	Matches one word in the list

```
$ ls -l /etc/host*
$ ls -l /etc/hosts.{allow,deny}
$ ls -l /etc/hosts.[!a]*
$ ls -l /etc/host?
```

15

Wildcards

- Command completion will not work after the first wildcard
- Multiple wildcards can be used
- `ls */*dat`

16

Navigating the filesystem: cd

18

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
    ```

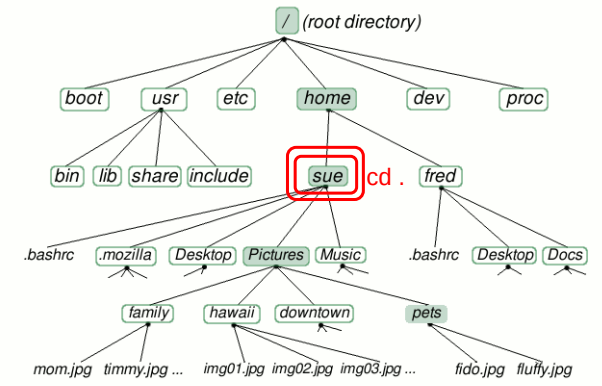
19

Directory shortcuts

- 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

20

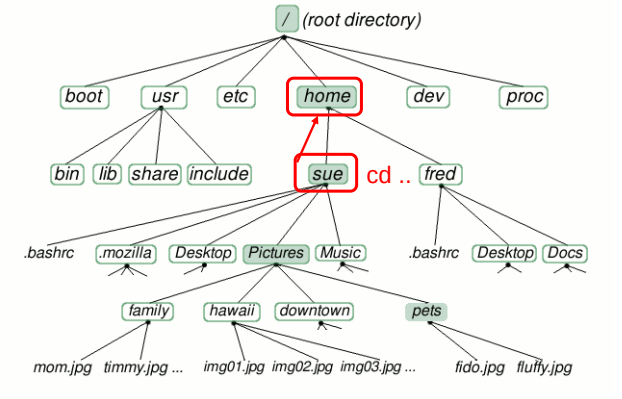
Linux File System - directories



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

21

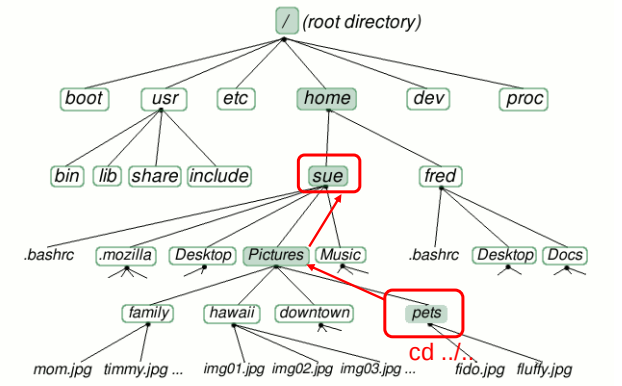
Linux File System - directories



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

22

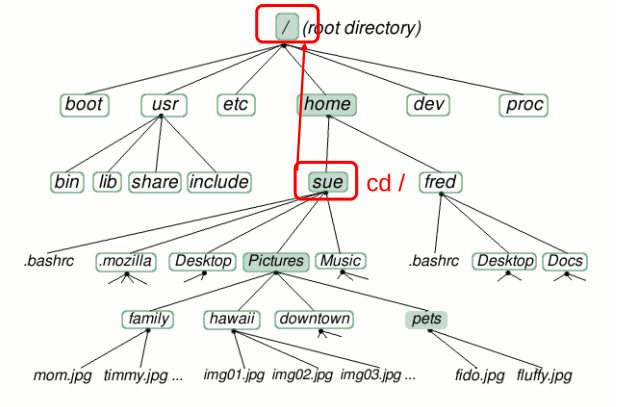
Linux File System - directories



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

23

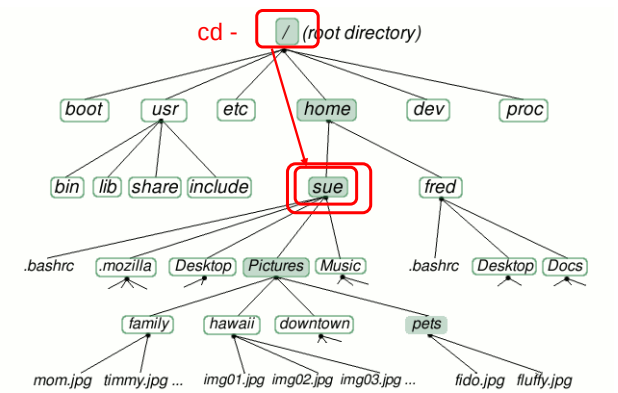
Linux File System - directories



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

24

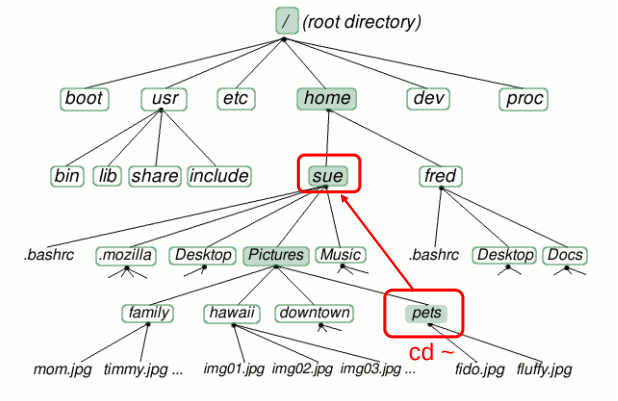
Linux File System - directories



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

25

Linux File System - directories

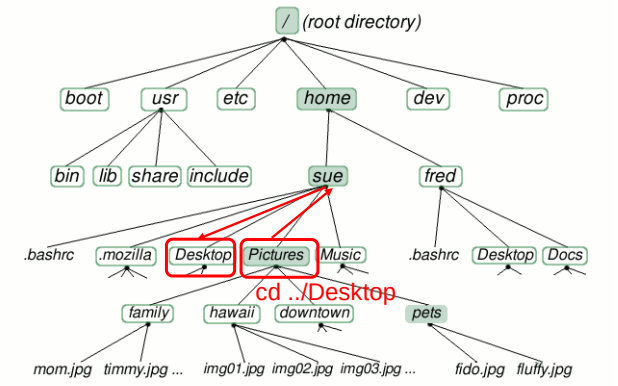


Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

26

Linux File System-home directory

Relative path

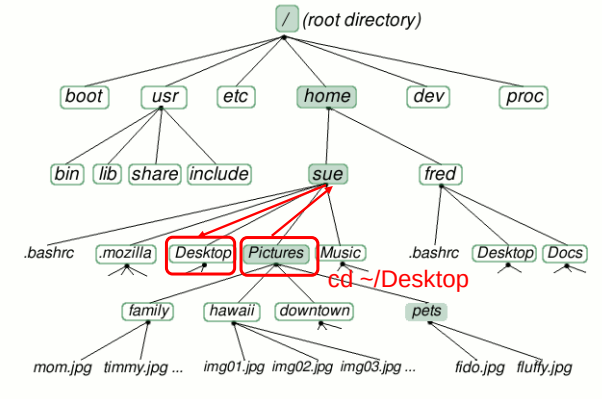


Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

27

Linux File System - directories

Relative path

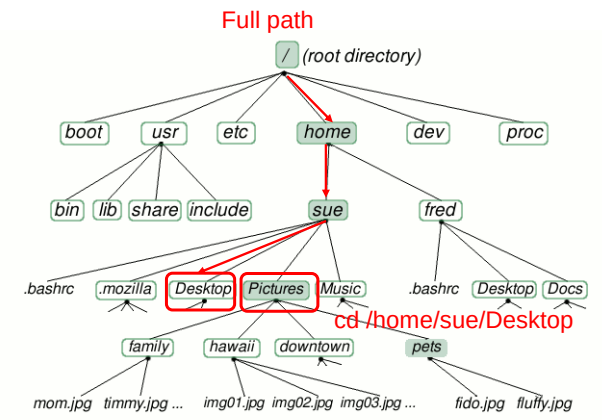


Source: http://www.linuxplanet.com/linuxplanet/tutorials/6666/1/screenshot3894/

28

Linux File System - directories

Full path



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

29

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

30