Introduction to Bioinformatics online course: IBT

Linux

Introduction to Linux and Unix and the command line
Learning Objectives

① Understand the Unix file structure
② Understand full vs relative paths: when and how to use them
③ Learn how to create directories and navigate through the file structure
④ Learn some useful shortcuts
Learning Outcomes

① Be able to create a file structure
② Be able to navigate through the file structure
③ Be able to create text files and view their content
④ Be able to use simple shortcuts
Introduction to Linux and UNIX
What is Linux?

- UNIX is an **Operating System (OS)** initially developed in the 1960.
- There are many different versions of UNIX, that share common similarities.
- The most popular varieties of UNIX are Solaris, **Linux** and MacOS.
- UNIX systems have a graphical user interface (GUI) making them easier to use.
Linux vs Unix

- Linux is a “clone” of the original Unix but doesn’t contain its code
- Linux is free and open source, the original Unix is not (but some of its derivatives are)
- All command lines work the same on both
Why Linux?

- Linux is free and the most popular distributions are Ubuntu, Fedora/Red Hat, Mandriva, etc.
- Low cost and very stable system
- Most secure OS
- Best multi-user and multi tasking OS
- The world’s fastest super computers run Linux
- Fast developing OS (many developers)

Very popular as servers OS
Linux distributions

- Different Linux distributions are available

- Ubuntu distribution is easy and convenient to use for beginners

- A simple guide to install Ubuntu in your machine:
Parts of the UNIX System

- The core operating system
- Accessible via the Terminal
- Programs and softwares

Adapted from: www.usna.edu
The Terminal

• A **terminal** refers to a wrapper program which runs a shell

• There are many different Unix shells, the most popular shell for interactive use include **Bash**: the default on most Linux installations
The Terminal

Shell prompt

User name
Machine name

Even though it is a command line interface, the mouse is still handy (scroll, copy, paste, etc.)
File-system under UNIX
Linux files structure

- **Root directory of the entire file system**
  - `/`
  - **home**
  - **dev**
  - **bin**
  - **etc**

**Users home directories**
- **Crick**
- **Watson**
- **Sanger**

**Configuration files**
- **Device files**
- **Essential user commands binaries**

**Watson files**
- **Desktop**
- **Documents**
- **IBT2017**

**IBT2017**
- **Genomics**
- **Linux**

- **Session 1**
- **Session 2**
Home directory and working directory

• When you first log in on a UNIX system, the working directory is your home directory.

• While working you will be associated to one directory called the working directory or the current directory

• An abbreviation of the working directory is displayed as part of the prompt on your terminal

• The command `pwd` gives the absolute path of the working directory
What is a path or a pathname?

• A path locates a given file in the system hierarchy
• An **absolute path** in the file system hierarchy for a given file or folder describes the parents all the way up to the root
• A **relative path** describes the path to the file starting from the **current working directory**
~ (your home directory)

- ~ refers to the home directory in a given file system
- The tilde ~ character can be used to specify paths starting at your home directory
The full path to file1 is:
/home/Watson/IBT2017/Linux/Session1/file1
Absolute path?

```
/home/Watson/IBT2017/Linux/Session1/file1
```

```
/home/Watson/IBT2017/Linux/Session1/file2
```
Every directory has two special sub-directories:

- (dot): the current directory
- .. (dot-dot): the parent directory
Relative path?

- `/`
- `home`
- `Watson`
- `Desktop`
- `Documents`
- `Genomics`
- `IBT2017`
- `Linux`
  - `/Linux`
  - `Session 1`
    - `./Linux/Session1`
    - `file1`
    - `file2`
  - `Session 2`
  - `./Linux/Session1/file1`

Current working directory refers to the current directory.
First test of the terminal

- Open the Terminal on your system
- The shell prompt will appear

Ok, let’s try some typing!
Creating directories and navigating through the file structure
## Commands for manipulating directories

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mkdir</code></td>
<td>Make directory: creates a new directory</td>
</tr>
<tr>
<td><code>rmdir</code></td>
<td>Removes a directory</td>
</tr>
<tr>
<td><code>pwd</code></td>
<td>Displays the absolute path of the current working directory</td>
</tr>
<tr>
<td><code>cd</code></td>
<td>Change directory: allows moving from one directory to another</td>
</tr>
<tr>
<td><code>ls</code></td>
<td>Lists a directory content</td>
</tr>
</tbody>
</table>
pwd command

• `pwd`: print working directory
• Displays the **absolute path** of your current location in the file system
• Try `pwd` on your terminal
• You should see: `/home/YourUsername`
Is command

• Is lists the content of the current directory by default
• Command structure `ls [OPTION] [dirname]`
• Some useful options:
  • `-l`: shows sizes, modified date and time, file or folder name and owner of file and permissions
  • `-a`: List all files including hidden file starting with ‘.‘
  • `-lh`: shows sizes in easier readable format
  • `-R`: recursively lists sub-directories
  • `-ls`: sorting by file sizes
Create a directory

- **mkdir**: makes a directory
- **Command structure**: `mkdir dirname [path]`

  - `mkdir dirname`: would create a directory with the specified `dirname`
  - The new created directory will be created in your current working directory
  - If you want to create it elsewhere, you have to specify the path: `mkdir dirname path`
Commands basic structure

command [–options] [arguments]

Example:

ls –lh /home/Watson/IBT2016
pwd
mkdir Test1
What you should know about file names in Linux

• No real distinction between the names of ordinary files and the names of directory files.
• No two files in the same directory can have the same name.
• Files in different directories can have the same name.
• Linux is case-sensitive: Sanger, sanger and SANGER are different and would represent three distinct files.
• In most cases, file extensions are optional (.txt, .exe, etc.)
Move in the files system

- **cd**: change the working directory
- Command structure: `cd <path>`
  - The path name of the directory you want to move to should be specified
  - You can specify either the absolute path or the relative path
Move in the files system: example

- **Move to Watson directory**
  1. `cd /home/Watson`
  2. `cd ../..
  3. `cd .. + cd ..`

- **Move to Genomics directory**
  1. `cd /home/Watson/IBT2017/Genomics`
  2. `cd ../Genomics`
  3. `cd .. + cd Genomics`
Remove a directory

• `rmdir`: removes a directory
• Command structure: `rmdir dirname [path]`
• It would remove the `dirname` directory
• The directory should be in your current working directory
• If you want to remove it from elsewhere, you have to specify the path: `rmdir dirname path to the directory`
• `rmdir` works if there is no contents in the directory
Remove a directory

- `rmdir` works if there is no contents in the directory
- If the directory contains files or sub-directories, an error message will appear: “Directory not empty”
- There is an option to remove `-r`, which stands for recursive, that will recursively remove a directory and its contents
How to get help for a command from the terminal?

• `man commandname` displays
Part 4

Some useful shortcuts and Links
Useful Links

• **cd**: cd followed by nothing will change the working directory to your home directory

• **cd ~user_name**: moves to the specified user home directory
Linux, some useful CLI key combinations

• **Ctrl+A**: move the cursor to the beginning of the command line
• **Ctrl+C**: end a running program and return to the prompt
• **Ctrl+D**: logout from the current shell session: equivalent to exit
• **Tab**: autocomplete a file name
• **Tab Tab**: displays command completion possibilities
• **Ctrl+L**: clear the terminal
Thanks

• Shaun Aron & Sumir Panji