Introduction to Bioinformatics online course: IBT

Linux

Manipulating files
Learning Objectives

① Learn how to create and edit files
② Learn how to view files content
③ Learn basic commands to manage files and directories
④ Learn some useful wildcards
Learning Outcomes

① Be able to create and edit files
② Be able to view files content
③ Be able to manage files and directories
④ Be able to use some wildcards
Creating new files
Basics manipulating file commands

- **touch** is used to create, change and modify timestamps of a file
- **touch** command creates an empty (zero byte) new file using this
  
  Structure: `touch filename`

- Create more than one single file
  
  `touch filename1 filename2 filename3`
touch command options

- **-a**: change the access time only
- **-c**: if the file does not exist, do not create it
- **-d**: update the access and modification times
- **-m**: change the modification time only
- **-r**: use the access and modification times of file
- **-t**: creates a file using a specified time
Text editors

- **nano**: a simple and easy-to-use text editor
- Is installed by default in Ubuntu and many other Linux distributions
- It’s a WYSIWYG editor: “what you see is what you get.” What you type directly goes into the text input
- **vim, emacs, gedit, Geany**: excellent programs but do require some learning
Get started with **nano**

- Name of the program and version number,
- The name of the file you are editing.

**shortcuts**

*Click to add text*
Let’s play with **nano**

- Open the terminal
- Log in (`/home/Username`)
- Type `pwd` to check
- Move to the directory Session1 directory under the Linux

```bash
cd Session1
```
Get started with **nano**

- **nano file1**
- Type “my first test file with webminal”
- Hit enter to move to another line and type “the second line of test”
- One you finish typing, hit Ctrl+x
- **Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES) ?**
- Hit Y
- **nano file2**
- Type “my second test file with webminal” and any other 4 lines of text
Some nano shortcuts

• To search for a text string, hit Ctrl+W, and enter your search term
• This search can then be cancelled mid-execution by hitting Ctrl+C without destroying your buffer
• Ctrl+X: finish typing and close an open file

Remember: nano pathname

- Opens the file if it’s existing already, you can modify and save changes
- Creates a new file in the specified path if it does not exist
Basic manipulating file commands
Displaying whole content of a file or parts of it \textit{(default + options)}

- **cat**: view the content of a short file  
  syntax \texttt{cat <filename>}
- **more**: view the content of a long file and navigate through it  
  syntax \texttt{more <filename>}
- **less**: view the content of a long file, by portions  
  syntax \texttt{less <filename>}
- **head**: view the first lines of a long file  
  syntax \texttt{head <filename>}
- **tail**: view the last lines of a long file  
  syntax \texttt{tail <filename>}
View file content: less command

- **less** command displays a text file content, one page at a time
- Structure: **less** filename
- Move a page down: either use the page down key or **space**
- To **exit** less, type **q**
- To go to the end of the text file, type **g**
Head and tail commands

- **head** command displays a text file content, by default: 10 first lines at a time
- Syntax: `head <options> <filename>`

- **tail** command displays a text file content, by default: 10 last lines at a time
- Syntax: `tail <options> <filename>`
Basic manipulating file commands

- **Copy, move and remove**
  - **cp**: copy files and directories
    Structure: `cp <path from> <path to>`
  - **mv**: move or rename files and directories
    Structure: `mv <path from> <path to>`
  - **rm**: remove files and directories
    Structure: `rm pathname`
Copying command: `cp`

- Simplest form: `cp file1 file2`
  
  Copy the contents of file1 into file2. If file2 does not exist, it is created. Otherwise, file2 is silently overwritten with the contents of file1.

- `cp filename dirpath`
  
  Make a copy of the file (or directory) into the specified destination directory
Other examples: `cp`

- Add the interactive mode with the option `-i`
  
- `cp -i file1 file2`
  
  ➔ Same as the previous one. However, if file2 exists, the user is notified before overwriting file2 with the content of file1

- `cp –R pathdir1 pathdir2`
  
  ➔ Copy the contents of the directory `dir1`. If directory `dir2` does not exist, it is created. Otherwise, it creates a directory named `dir1` within directory `dir2`
Copying command: `mv`

The `mv` command moves or renames files and directories depending on how it is used.

- **To rename a file:**
  
  \[ mv \text{ filename1} \text{ filename2} \]

  If file2 exists, its contents are silently replaced with the contents of file1. To avoid overwriting, use the interactive mode:
  
  \[ mv -i \text{ filename1} \text{ filename2} \]

- **To move a file (or a directory) to another directory:**
  
  \[ mv \text{ file} \text{ dirpath} \]

- **To move different files (or a directory) to another directory:**
  
  \[ mv \text{ file1 file2 file3} \text{ dirpath} \]

- **To move directory to another directory:**
  
  \[ mv \text{ dir1} \text{ dir2} \]

  If dir2 does not exist, then dir1 is renamed dir2. If dir2 exists, the directory dir1 is moved within directory dir2.
The **`rm`** command

The **`rm`** command deletes files and directories

**To remove a file:**

```
rm filename
```

**To remove many files:**

```
rm filename1 filename2
```

Add the interactive mode to prompt user before deleting with **`-i`**

```
rm -i filename1 filename2
```

**Delete directories with all their contents**

```
rm -r dir1 dir2
```
Be careful with rm!

- Linux **does not have an undelete command**
- Once you delete something with `rm`, it's gone!
- You can inflict terrific damage on your system with `rm` if you are not careful, particularly with wildcards
- Try this trick before using `rm`: **construct your command using `ls` instead first**
Wildcards

• Since the shell uses filenames so much, it provides special characters to help rapidly specifying groups of filenames

• A group of special characters are called wildcards allow selecting filenames based on pattern of characters
# Wildcards

<table>
<thead>
<tr>
<th>Wildcard</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>Matches any characters</td>
</tr>
<tr>
<td>?</td>
<td>Matches any single character</td>
</tr>
<tr>
<td>![characters]</td>
<td>Matches any character that is not a member of the set characters</td>
</tr>
<tr>
<td>[characters]</td>
<td>Matches any character that is a member of the set characters. The set of characters may also be expressed as a POSIX character class such as one of the following:</td>
</tr>
<tr>
<td></td>
<td>[:alnum:] Alphanumeric characters</td>
</tr>
<tr>
<td></td>
<td>[:alpha:] Alphabetic characters</td>
</tr>
<tr>
<td></td>
<td>[:digit:] Numerals</td>
</tr>
<tr>
<td></td>
<td>[:upper:] Uppercase alphabetic characters</td>
</tr>
<tr>
<td></td>
<td>[:lower:] Lowercase alphabetic characters</td>
</tr>
</tbody>
</table>

Source: [http://linuxcommand.org](http://linuxcommand.org)
## Wildcards examples

<table>
<thead>
<tr>
<th>Wildcard</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>a*</td>
<td>Any file name starting with a</td>
</tr>
<tr>
<td>*</td>
<td>All possible filenames</td>
</tr>
<tr>
<td>A*.fasta</td>
<td>All filenames that begin with A and end with .fasta</td>
</tr>
<tr>
<td>????.vcf</td>
<td>Any filenames that contain exactly 4 characters and end with .vcf</td>
</tr>
<tr>
<td>[abc]*</td>
<td>Any filename that begins with &quot;a&quot; or &quot;b&quot; or &quot;c&quot; followed by any other characters</td>
</tr>
<tr>
<td>[:upper:]*</td>
<td>Any filename that begins with an uppercase letter. This is an example of a character class</td>
</tr>
</tbody>
</table>
Download files from the web

- **wget** stands for "web get". It is a command line utility which downloads files over a network.
- It supports HTTP, HTTPS, and FTP protocols.

**Syntax:** `wget [–options] [URL]`

**Let’s try it:**
- Move to the directory Genomics and get the fasta file of P. falciparum from PlasmoDB.
- **Command:** `wget http://plasmodb.org/common/downloads/release-9.0/Pfalciparum/fasta/PlasmoDB-9.0_Pfalciparum_BarcodeIsolates.fasta`
Thanks
Shaun Aron & Sumir Panji