

# First steps with Unix - Lab

**Boris Steipe**

boris.steipe@utoronto.ca

<http://biochemistry.utoronto.ca/steipe>

Department of Biochemistry  
Department of Molecular and Medical Genetics  
Program in Proteomics and Bioinformatics  
University of Toronto

Laboratory 1.2a



<http://creativecommons.org/licenses/by-sa/2.0/>



The image is a yellow rectangular graphic with a black border, containing the Creative Commons Attribution-ShareAlike 2.0 license summary. At the top center is the Creative Commons logo (two 'C's in a circle) followed by the text 'creative commons' in a bold, sans-serif font, and 'COMMONS DEED' in a smaller, spaced-out font below it. Underneath is the title 'Attribution-ShareAlike 2.0'. The text is organized into sections: 'You are free:' followed by a bulleted list of permissions; 'Under the following conditions:' followed by two icons in circles (a 'BY' and a circular arrow) with their respective descriptions; a bulleted list of additional conditions; a statement about fair use; and a link to the full legal code with a disclaimer icon.

**creative commons**  
COMMONS DEED

**Attribution-ShareAlike 2.0**

**You are free:**

- to copy, distribute, display, and perform the work
- to make derivative works
- to make commercial use of the work

**Under the following conditions:**

**BY:** **Attribution.** You must give the original author credit.

**Share Alike.** If you alter, transform, or build upon this work, you may distribute the resulting work only under a license identical to this one.

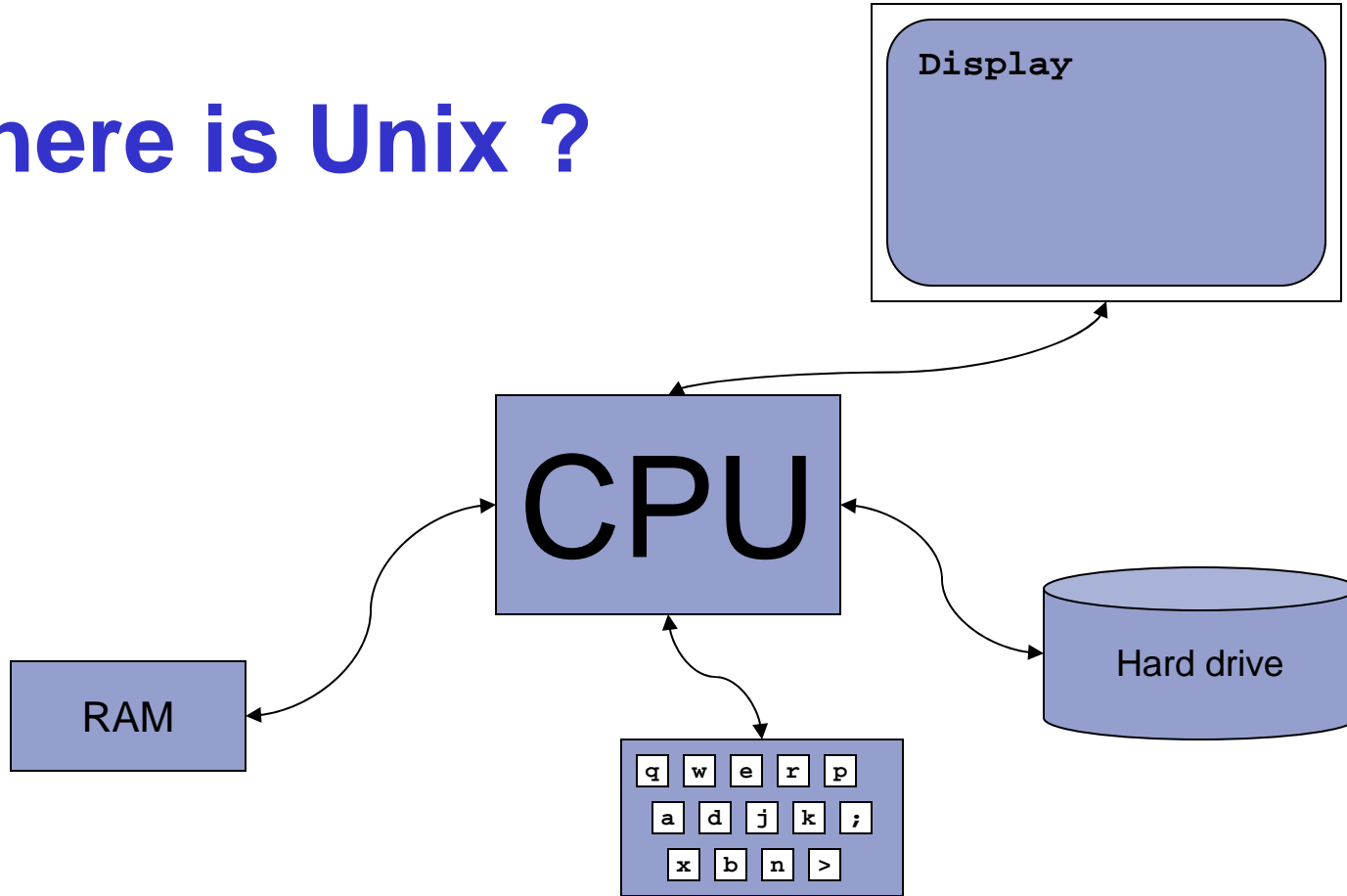
- For any reuse or distribution, you must make clear to others the license terms of this work.
- Any of these conditions can be waived if you get permission from the copyright holder.

**Your fair use and other rights are in no way affected by the above.**

This is a human-readable summary of the [Legal Code \(the full license\)](#).

[Disclaimer](#) 

# Where is Unix ?

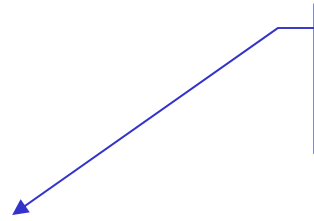
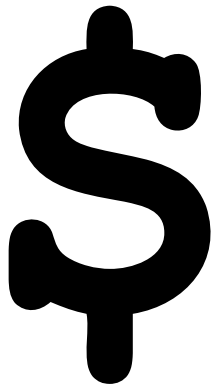


Unix is an **operating system**. It is a program that runs on a computer and organizes the way the computer works. It's job is mostly behind the scenes. **Except ...**

... when we interact directly with a "shell".

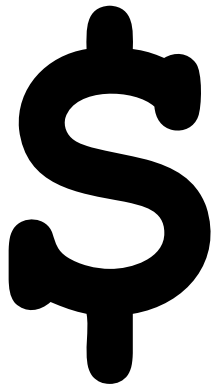
```
user: steipe
password:
$
```

# The "shell"



**This is real Unix !!!**

# The "shell"



(Actually this is the shell's prompt. Ready to take your orders.)

```
$ echo $SHELL  
/bin/bash
```

# Prompts

Prompts give feedback on who/where you are ...  
Different flavors:

**Simple**

**MacOS X**

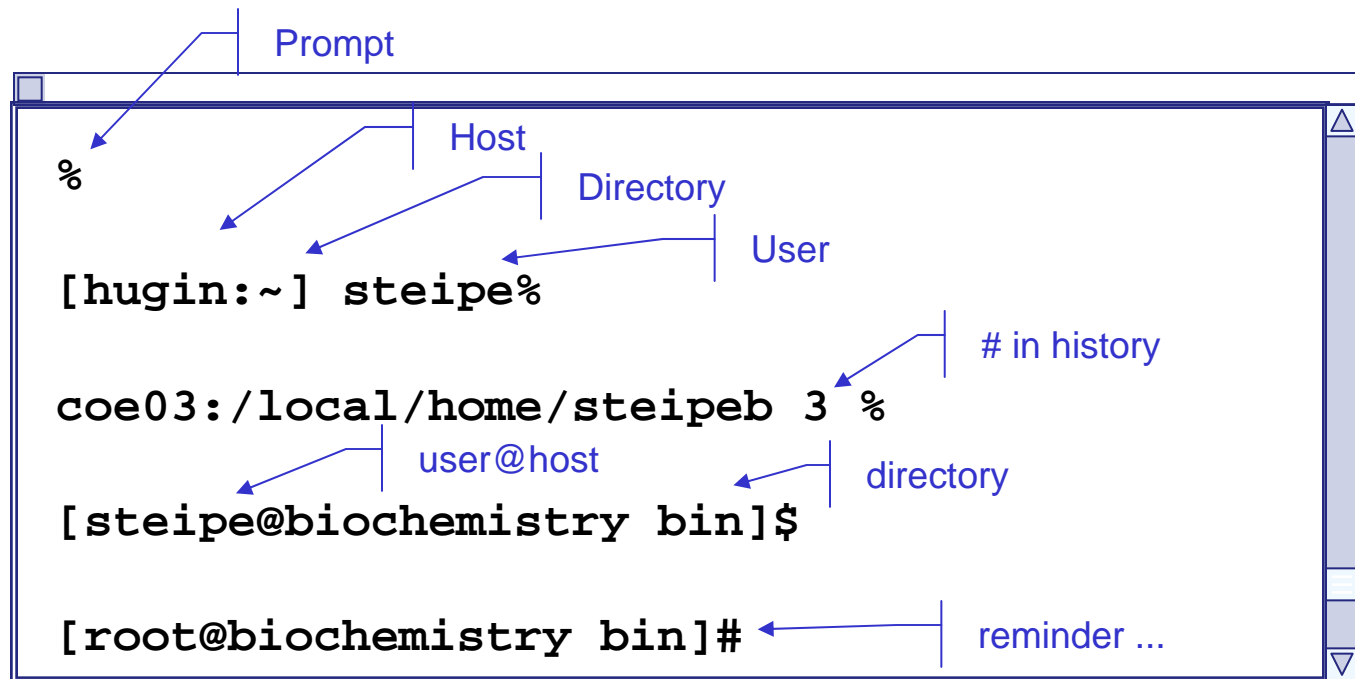
(tcsh)

**Solaris**

(tcsh)

**Linux**

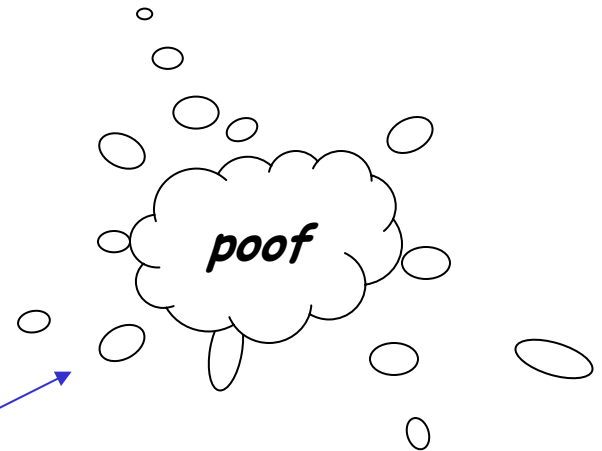
(bash)



you can change the prompt (eg tcsh): set prompt = ...

# exit

Terminate a shell process and its associated subprocesses.  
Close a window. Log out.



Usually. Except if there are stopped jobs. These you need to terminate first.

# Basic Unix commands

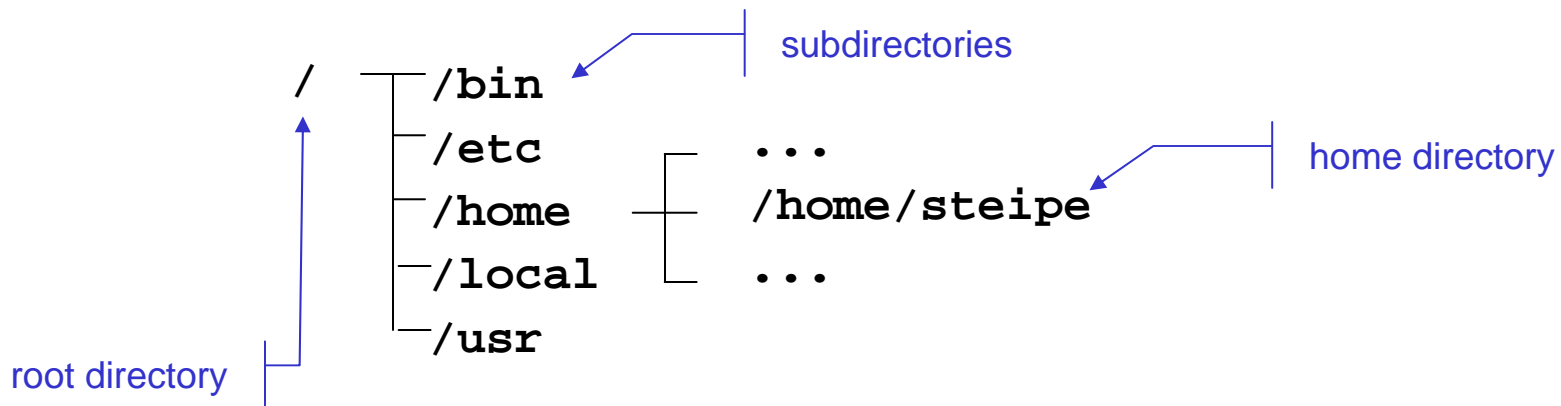
- cd** – change directory
- ls** – list
- cp** – copy a file
- mv** – move or rename a file
- rm** – remove a file (use caution)
- mkdir** – make a directory
- rmdir** – remove a directory (must be empty)
- pwd** – print working directory

... see Lab notes

# Who/where are we anyway ?

```
$ pwd
/home/steipe
$ whoami
steipe
```

## Unix directory structure:



# cd

## change directory

```
/home/steipe$ ls -l
total 4
drwxr-xr-x  2 steipe  wheel   512 Nov 30 01:09 project
-rwx---r--  1 steipe  wheel    93 Nov 28 17:01 test.pl

/home/steipe$ cd project
/home/steipe/project$ ls
test1.pl    1JKZ.pdb

/home/steipe$ cd ..      (change to parent directory)
/home/steipe$ cd /       (change to root directory)
/home/steipe$ cd         (change to home directory)
```

! |

← Or enter any full path

# Directories worth knowing

**/home / <user>**

: home directory

**/etc**

: system resources

**/usr/bin**

: locally installed applications

**/usr/local/apache/htdocs** : http root directory

# man

## manual pages

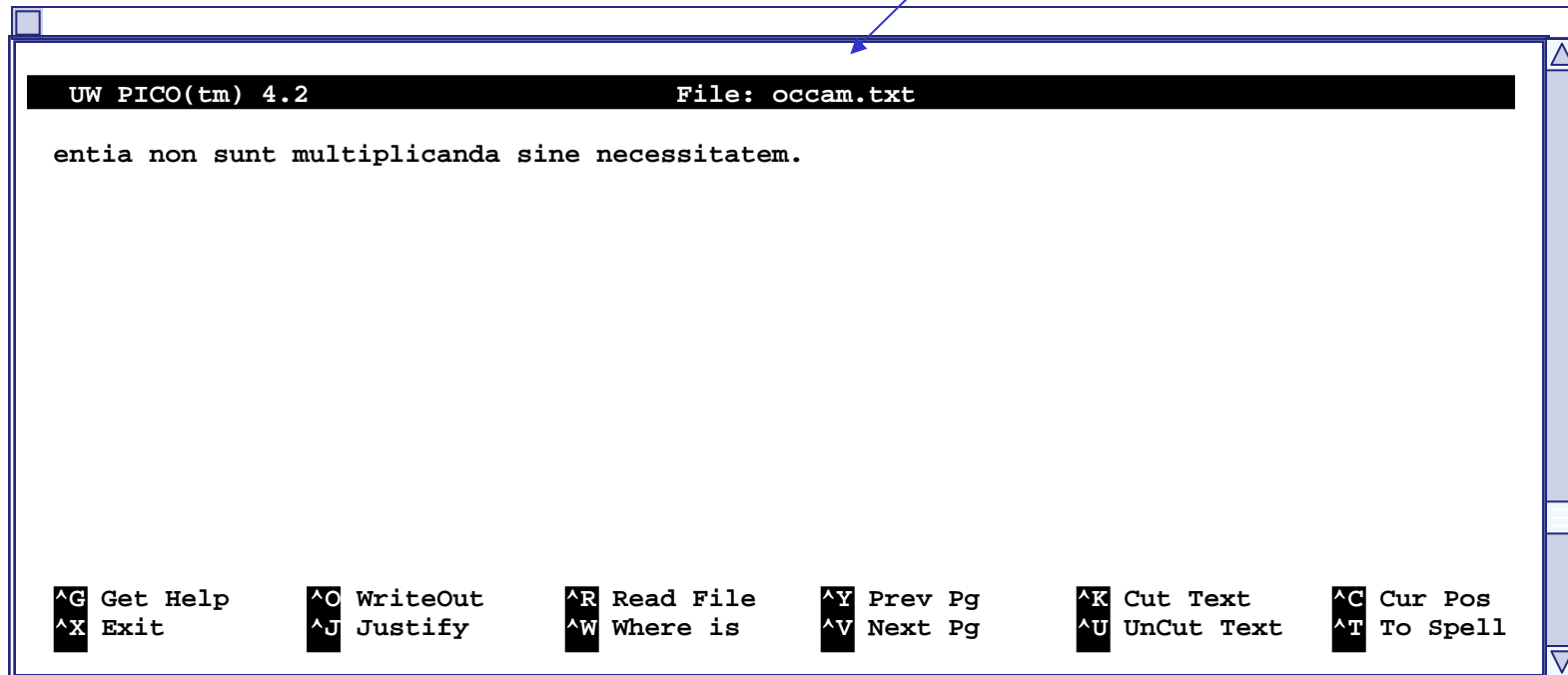
```
% man <command>
% man -k copy ← searches relevant
% man cp      entries by keyword
[...]
```

**<spacebar> to page**  
**q to quit**

# pico/nano

## A screen based text editor

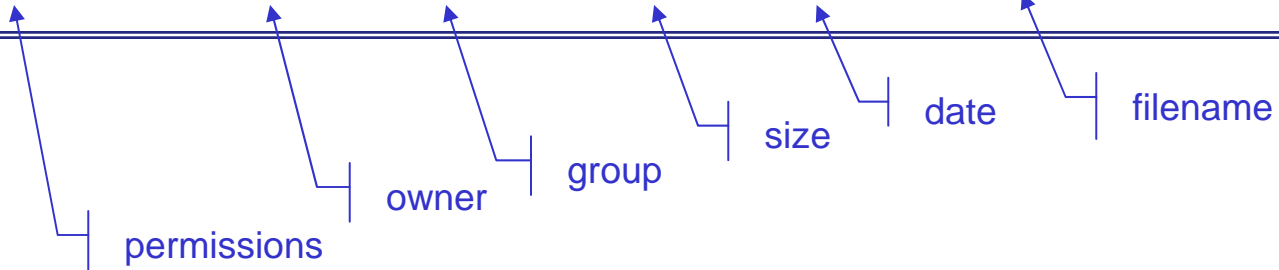
invoke with: `pico <filename>`  
(Dont use special characters  
or blank spaces in filenames.)



# ls

## list

```
home/steipe/project$ ls
occam.txt          1JKZmdl1.pdb
home/steipe/project$ ls -l
total 2354
-rw-r--r--   1 steipe  wheel  1125819 Nov 30 00:57 1JKZ.pdb
-rw-r--r--   1 steipe  wheel    49 Jan 25 00:13 occam.txt
home/steipe/project$ ls -la
total 2358
drwxr-xr-x   2 steipe  wheel   512 Nov 30 01:09 .
drwxr-xr-x   7 steipe  wheel   512 Nov 30 01:09 ..
-rw-r--r--   1 steipe  wheel  1125819 Nov 30 00:57 1JKZ.pdb
-rw-r--r--   1 steipe  wheel    49 Jan 25 00:13 occam.txt
```



# List: some commandline flags

## **ls**

- List files in current directory

## **ls -F**

- Lists files with decoration for file type

## **ls -l**

- Lists extra information about each file (size, ownership, permission status, and date)

## **ls -al**

- Lists all files including hidden files in directory and tells you information about each

## **ls -t**

- Lists files in order according to the date

## **man ls**

- Tells you what ls can do

# mkdir

make a new **directory**

```
$ cd  
$ mkdir test  
$
```

# cp

## copy

```
$ cp ../steipeb/ACGC2/1JKZmdl1.pdb 1JKZmdl1.pdb
$
```

Note: Always remember **cp <from> <to>**

Note: cp cheerfully overwrites without warning.

# rm

## remove

```
$ cp JKZ1mdl1.pdb test.txt
$ rm test.txt
$
```

**rm will cheerfully delete everything you tell it.  
Options, like rm -r are especially dangerous.**

```
$ alias rm='rm -i' ← safeguard: "interactive"
$
```

To remove directories: empty them first, then **rmdir**

# cat

concatenate and display

```
$ cat sparkle  
perl -e "while(){print rand>0.002?' ':'*'}"  
$
```

# more, head, tail

## commands to look at parts of files

```
$ more 1JKZmdl1.pdb  
[...]  
$ head 1JKZmdl1.pdb  
[...]  
$ head -50 1JKZmdl1.pdb  
[...]  
$ tail -15 1JKZmdl1.pdb
```

# a program

type ...

```
$ perl -e 'print "Wheeeeeee ... \n"'
```

# a "Happy Birthday" program

type ...

```
$ perl -e "while(){print rand>0.002?' ':'*'}"
```

# which

**which** lists the files that would be executed if the argument were entered as a command

```
$ which perl
/usr/bin/perl

$ which cp
/bin/cp
```

# Permission: a dissection

-rwxrwxrwx	Everyone can read [r], write [w] and execute [x] this file
-rw-----	Only the user can read and write this file
drwxr-xr-x	user can list [r], add or delete [w], and read files [x], in this <b>directory</b> . The group and other can only list and read.

<u>User</u>	<u>Group</u>	<u>Other</u>
rwx	rwx	rwx

# Changing permissions

- **chmod** - change the permissions mode of a file or directory
- **chmod a+rw <file>**
  - Gives everyone [r]eading and [w]riting privileges for <file>
- **chmod go-w+x <dir>**
  - Removes writing access and makes <dir> e[x]ecutable for [g]roup and [o]ther
  - Note: if <dir> is a directory [x] gives reading permission and [r] merely allows you to list its contents

symbol	meaning
u	user
g	group
o	other
a	all
+	give rights
-	remove rights

# access privileges (the generic way)

```
chmod 755 test.pl
```

Permissions	Binary	Octal
---	000	0
--x	001	1
-w-	010	2
-wx	011	3
r--	100	4
r-x	101	5
rw-	110	6
rwx	111	7

**Example:**

Octal:

**755**

Binary: 111 101 101

Meaning: rwx r-x r-x

&

Executes a program in the *background*

```
$ mozilla &  
$ ls  
[...]
```

# ps, jobs, fg

**ps** gives details on active jobs, **jobs** lists suspended jobs, **fg** takes them to the foreground

```
$ perl -e 'while(){print rand>0.002?" ":"*"}'  
[ ... happiness unfolds]  
  
^Z  
$  
$ ps  
  PID  TT  STAT      TIME COMMAND  
  5171  p1  S        0:00.69 -bash  
  5971  p1  S        0:00.91 perl -e while(){print rand>0.002?" ":"*"}  
  5714  p2  S        0:00.08 -bash  
$ jobs  
[1]+  Stopped                  perl -e 'while(){print rand>0.002?" ":"*"}'  
$ fg  
[ ... happyyness continues to unfold]  
<ctrl> c  
$
```

# kill

## Terminate a running process

```
$ perl -e 'while(){print rand>0.002?" ":"*"}'  
[ ... happiness unfolds]  
^Z  
$  
[1]+  Stopped                  perl -e 'while(){print rand>0.002?" ":"*"}'  
$ ps  
  PID TT  STAT      TIME COMMAND  
 5171 p1  S        0:00.70 -bash  
 5990 p1  S        0:01.09 perl -e while(){print rand>0.002?" ":"*"}  
$ kill -9 5990  
$ ps  
  PID TT  STAT      TIME COMMAND  
 5171 p1  S        0:00.71 -bash  
[1]+  Killed                    perl -e 'while(){print rand>0.002?" ":"*"}'  
$
```

# End of this section