

# Introductory Linux Tutorial for Life Sciences

## Session 1: Unix, Linux and the command line

Teacher: Marcel Martin



# Linux

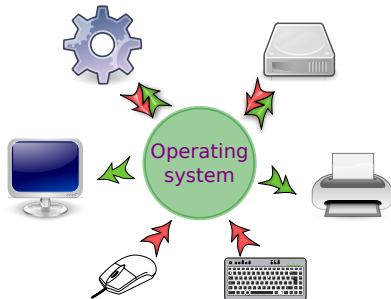
Linux is a Unix-like operating system kernel

# Linux

Linux is a **Unix-like operating system kernel**

# Operating system (OS)




- “A software that manages hardware and software resources and provides common services for programs” (WP:Operating system)
- The OS starts when the computer is switched on



# Components of an operating system

- The **kernel** has exclusive control over the computer's resources (processor, memory, input- and output devices)
- **Utilities** are small programs that help with managing the system
- **User interface** allows interaction with the machine
  - Graphical user interface (GUI)
  - Command-line interface (CLI)

# Unix

- An operating system conceived in the 1970s at AT&T Bell Labs
- Ancestor of many operating systems in use today:
  - macOS 
  - Solaris 
  - Linux 

# Linux

- A free, Unix-like operating system kernel
- Published in 1991 by Linus Torvalds (University of Helsinki)
- It's Free Software: Source code free for all to copy, study, change and share
- Used nearly everywhere
  - Supercomputers
  - PCs
  - Android smartphones
  - TVs
  - Video game consoles



# Distributions

- Linux *distributions* bundle the Linux kernel with other software into a usable operating system
- Examples: Debian, Fedora, SUSE Linux, CentOS, ...



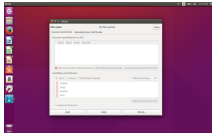
# Clarifications

- “Linux” is technically only the kernel
- But we often say “Linux” when we mean the entire system
- ... and much of what we say about Linux also applies to any other Unix

# User interfaces

- An *interface* is a shared boundary across which computer components exchange information

**GUI (graphical  
user interface)**



**CLI (command  
line interface)**



# Command-line interface

- User interacts with the computer by typing in instructions (commands)
- A text-based “conversation” between user and computer

Why? I want to use the mouse!

- Uses fewer resources
- Works remotely
- The only way to access some compute clusters
- Many programs are command-line only
- More powerful for certain tasks – after you’ve learned it

# The shell

- The program that lets you type in and run commands
- We will use Bash (the *B*ourne *A*gain *s*hell)
- There are others: sh, csh, tcsh, zsh<sup>1</sup>

Good to know:

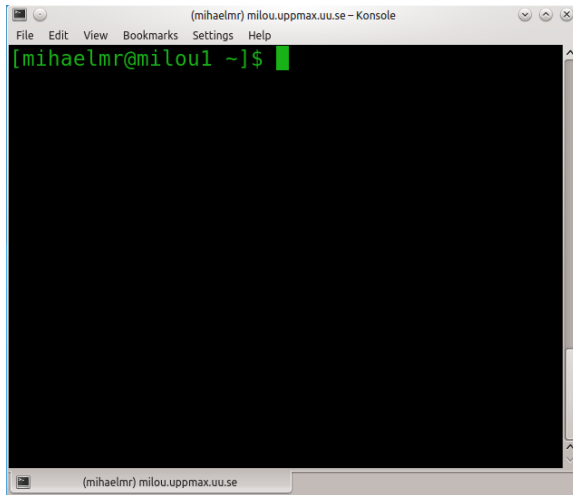
- Even Windows has shells: cmd.exe and PowerShell
- Running macOS? You have Bash

---

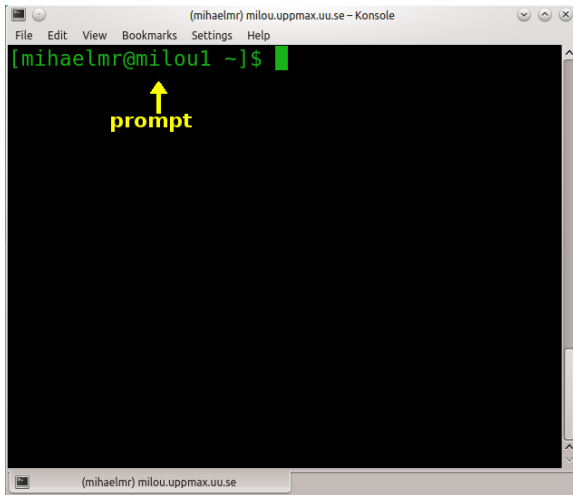
<sup>1</sup>Bonus question: Why are these names so short?

Start your web terminal now if you haven't

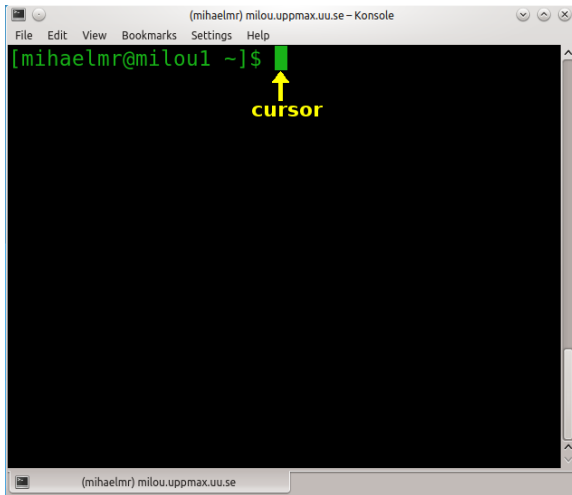
# The terminal



# The terminal

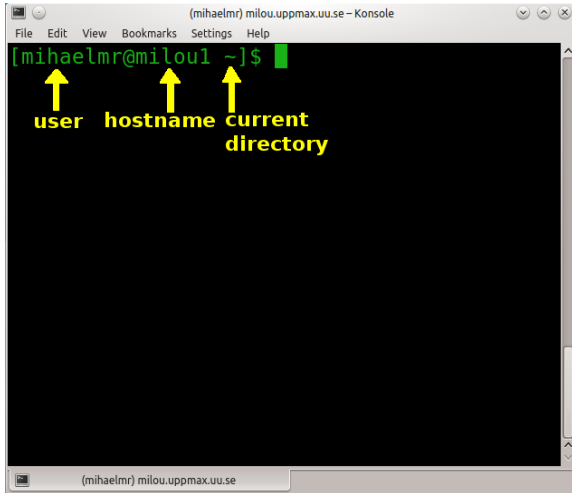


# The terminal





# The terminal



The image shows a terminal window titled "(mihaelmr) milou.uppmx.uu.se - Konsole". The terminal displays the prompt `[mihaelmr@milou1 ~]$` in green text. Three yellow arrows point from labels below to parts of the prompt: "user" points to "mihaelmr", "hostname" points to "milou1", and "current directory" points to "~". The terminal window has a menu bar with "File", "Edit", "View", "Bookmarks", "Settings", and "Help". The status bar at the bottom shows "(mihaelmr) milou.uppmx.uu.se".

# Commands

A *command* is an instruction typed in at the command line and processed by the shell. Example:

```
echo Hello World
```

- The first “word” is the *command (name)*<sup>2</sup>
- The remaining “words” are the *arguments*
- They are separated by whitespace
- The meaning of the arguments varies from command to command.

---

<sup>2</sup>Note ambiguity: “Command” can be the entire thing or just the initial part

# Options

- Arguments preceded by “-” are called *options* (or switches).
- They are used to change a command’s behavior:

```
$ ls
aFile.txt
$ ls -l
total 3
-rw-r--r-- 1 teacher1 teachers 197 Dec  4 09:28 aFile.txt
```

→ `ls -l` displays extra information

Try it now!

# Command-line syntax conventions

Documentation (often) describes how to use a command like this:










```
command [options] [-r <email>] { --this | --that }  
        argument [arguments ...]
```

- Parts enclosed in [ ] are optional
- Dots indicate that there can be more than one
- A <placeholder> needs to be replaced with something useful


# Types of commands

- *Internal* (or *builtin*): The shell knows what to do. Example: `echo`, `exit`
- *External*: A program somewhere on the disk. Example: `fastqc`, `samtools`
- Most commands are external

# Keyboard shortcuts

-  ,  – navigate in command history
-  ,  – move the cursor back/forth along the current line
-  a,  e – move the cursor to start/end of the line
-   /  – move from one word to another

# Autocompletion

- Write the first letters of a command or file name and press  (the tab key).
- If possible, the rest of the name is filled in automatically
- If nothing or only some letters are autocompleted, there are multiple possible completions
- Press tab a second time to see the possible completions
- Use this as often as possible! It saves time and avoids typos.

# First commands

Test a few commands. What do they do?

Note: Commands and file names are **case sensitive**

(`image.jpg`  $\neq$  `Image.JPG`)

- `date`
- `who`
- `echo`
- `ls`

Try out the commands, and also use some keyboard shortcuts!



# First commands

Test a few commands. What do they do?

Note: Commands and file names are **case sensitive**  
(`image.jpg`  $\neq$  `Image.JPG`)

- `date` – print the operating system date and time
- `who` – determine the users logged on the machine
- `echo` – print text
- `ls` – list files in the current directory

Try out the commands, and also use some keyboard shortcuts!

# Getting help

Getting help for a command:

`help <command>` – works if it is a builtin command


`man <command>` – works if a **manual** page exists

`command --help` or `command -h` – works if the command was programmed to know about that option

# Command: `man` – display manual pages

- Usage: `man <command>`

```
$ man whoami
NAME
  whoami - print effective userid
SYNOPSIS
  whoami [OPTION] ...
DESCRIPTION
  Print the user name associated with the current
  effective user ID
  --help display this help and exit
  --version output version information and exit
AUTHOR
  Written by Richard Mlynarik
```

- Exit with 

# Command: `help` – show help for builtins

- Displays brief summaries of shell builtin commands
- Usage: `help <command>`

```
$ help pwd
pwd: pwd [-LP]
    Print the name of the current working directory.

Options:
  -L          print the value of $PWD if it names the
              current working
              directory
...

```

# Command --help

- Usage: `<command> --help`

```
$ whoami --help
Usage: whoami [OPTION]...
Print the user name associated with the current
effective user ID.
Same as id -un.
    --help      display this help and exit
    --version   output version information and exit
```

# Summary

- **Unix** is a family of operating systems (OS)
- **Unix/Linux** OS can manage multiple users, multiple tasks, and networking
- The **shell** (Command Line Interpreter) is a program that reads commands typed into a terminal and executes them
- A **command** is an instruction typed in at the command line and processed by the shell