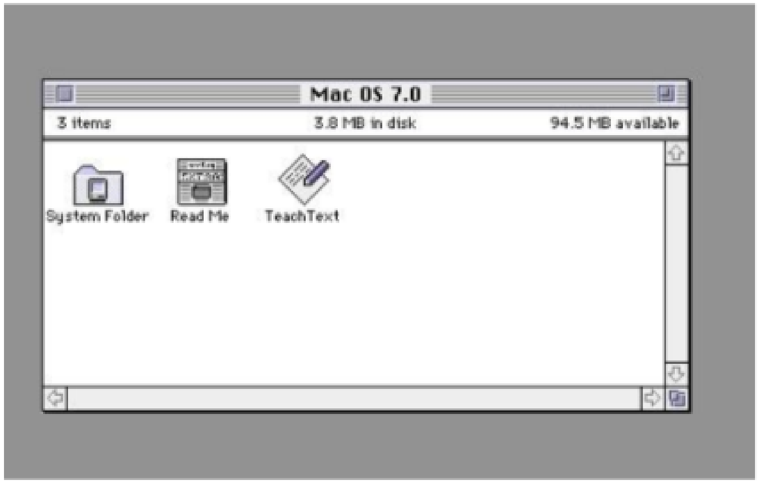
Introduzione all'uso di bash su Linux

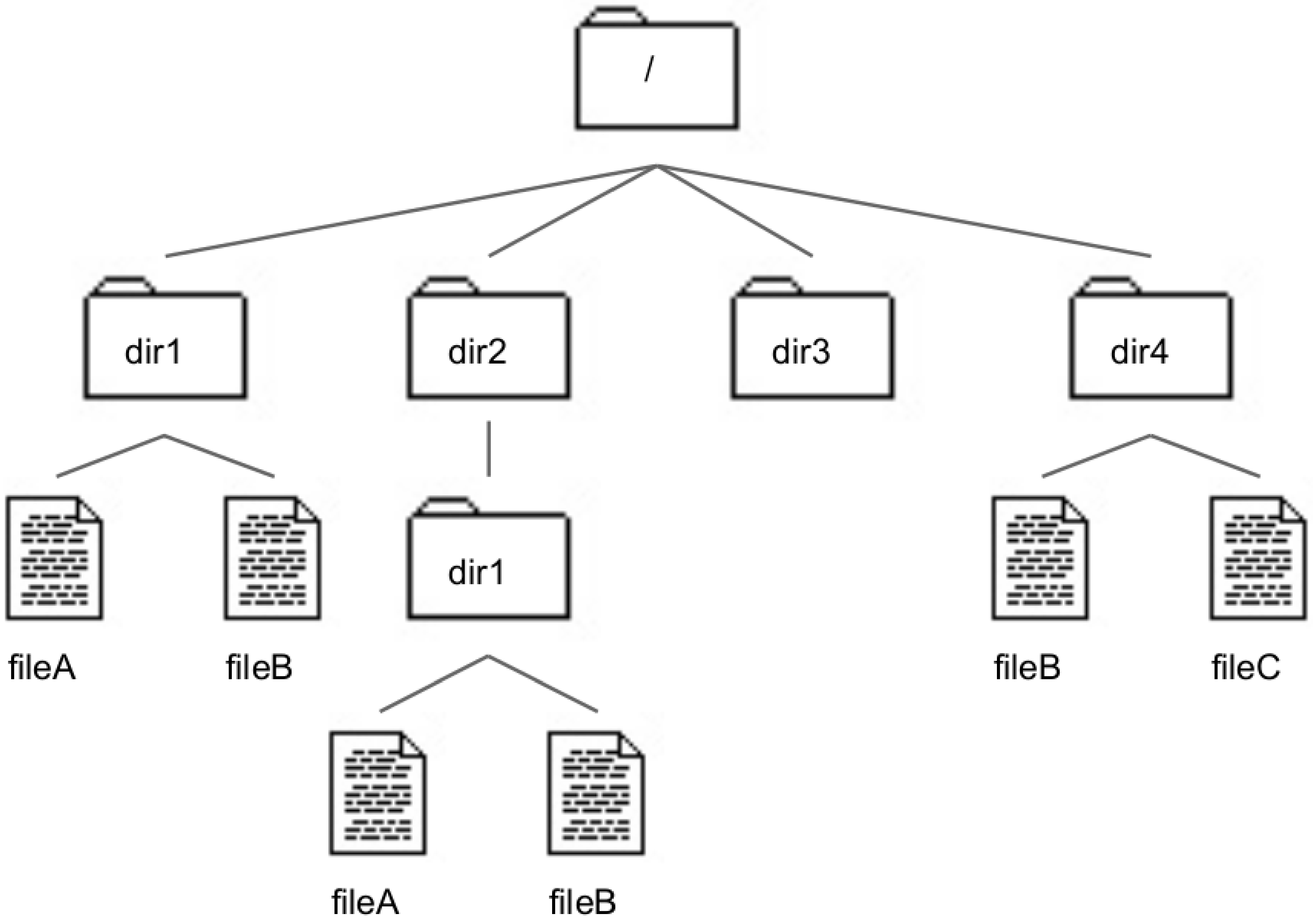
Shells

- textual shells (CLI)
  - unix shells, dos prompt

- graphic shells (GUI)
  - windows explorer, mac finder, linux desktop environments



How files are organized on disk



Browsing the filesystem

- a shell always has a working directory

- you can see it with “pwd”
- you can see the files in it with “ls”
- you can change it with “cd”

- at the beginning it is your home

Special names for special directories

- the working (current) directory:

  “.”

- the parent of the current directory:

  “..”

- the root directory:

  “/”

- the home directory:

  “~”

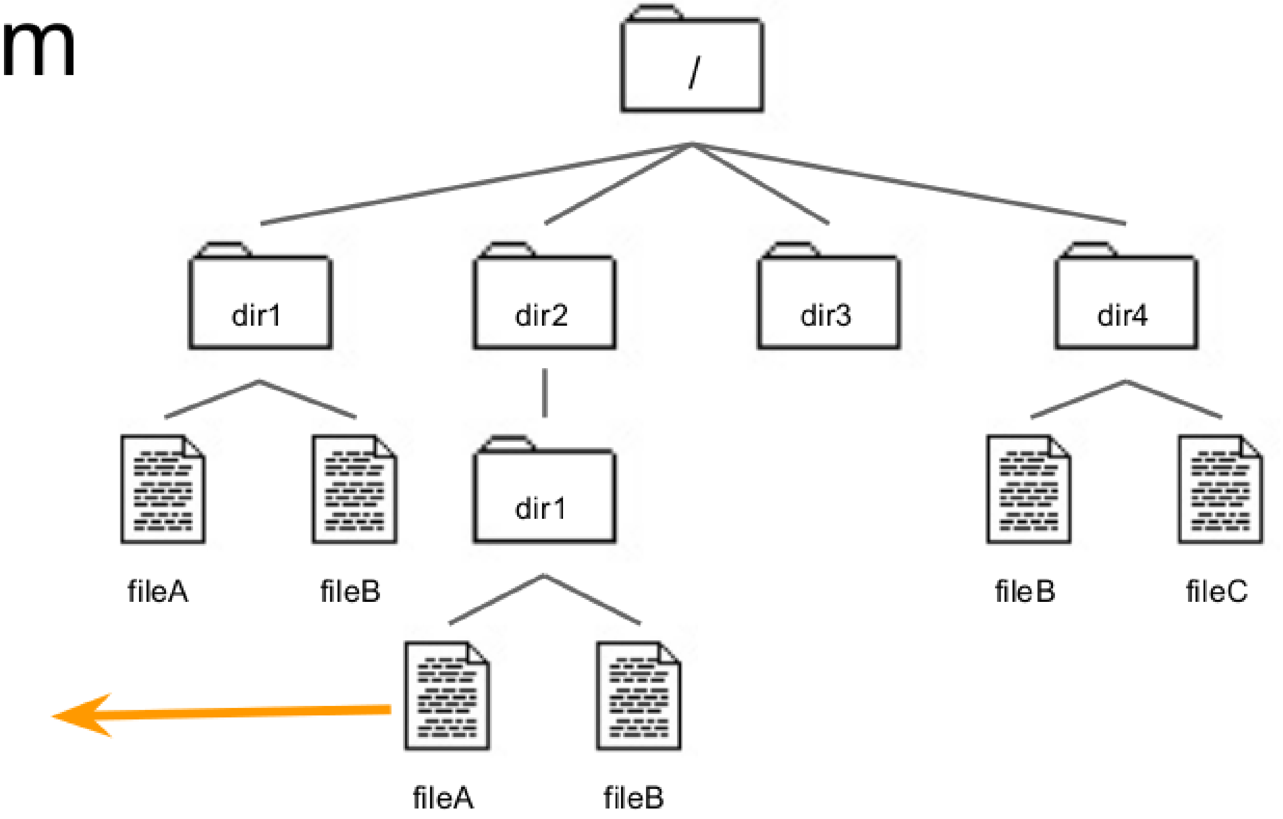
Paths

- absolute path: starts from `/`
- relative path: starts from the current directory

Absolute paths

Always begin from the root (/)

```
/dir1/fileA
/dir2
/dir4/fileB
/dir2/dir1/fileA
```



Relative paths

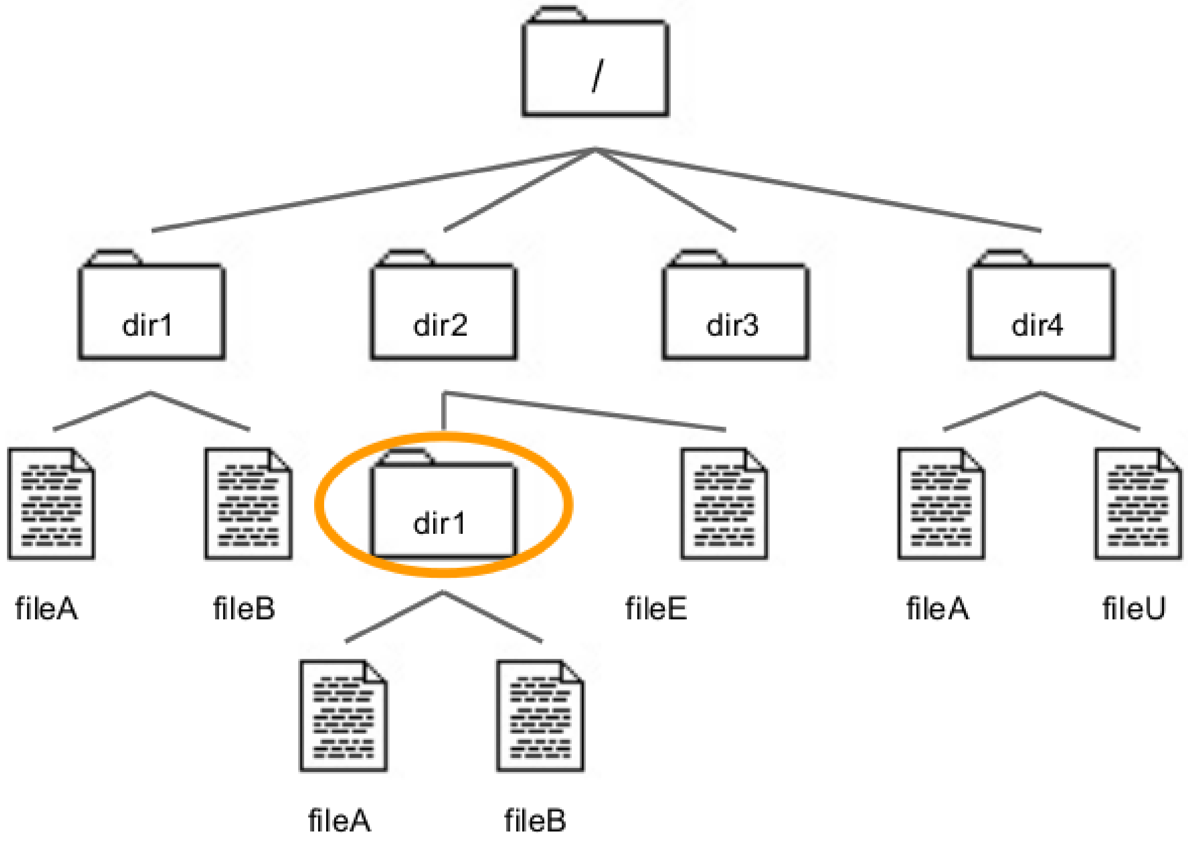
```
$ pwd
/dir2/dir1
```

- Need to use fileU?

```
../../dir4/fileU
```

- And fileE?

```
../fileE
```



Other commands

- Copy a file:

```
$ cp fileA copyOfFileA
```

- Rename a file:

```
$ mv fileA fileB
```

- Create a directory:

```
$ mkdir dir2
```

If you do not know how to use a command try:

```
$ command -h
$ command --help
$ man command
```

- You can Google it
- If everything fails ask us!

Command’s Anatomy

- Some commands are fine alone:

```
$ pwd
$ ls
```

- Many require one or more arguments:

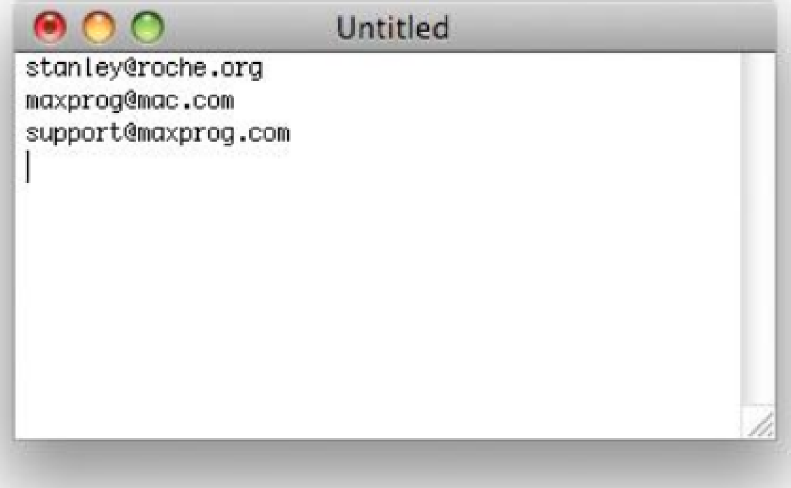
```
$ mkdir some_dir
$ mv a_file new_name
```

- Most commands can have options:

```
$ ls -l -a
```

Simple text files are your friends!

Simple text files contains text and nothing else.



```
stanley@roche.org
maxprog@mac.com
support@maxprog.com
```

- MP3 are binary files
- DOC, XLS and JPG are binary too!

- In Windows you can create text files with Notepad (not Word or Wordpad!)

Bioinformatics and Text files

- Most bioinformatics format are simple text files:
    - Fasta
    - FASTQ
    - SAM
    - VCF
    - BED

primers.fa

```
>beta_lac_FOR
ACGATCGTTACGTACTTGGGGGG
>beta_lac_REV
ACGTACGTATTTTTACGATCGATC
>ctrl_FOR
CAGCTATTATCGTATCGTACGACT
>ctrl_REV
CAGCAGCTACGTACGGTAGCATGC
```

How to see the content of a file

- The command `cat` prints the content of a text file

```
$ cat primers.fa
```

- The command `wc` prints the number of lines

```
$ wc -l primers.fa
```

- The command `head` prints the first lines

```
$ head primers.fa
```

- The command `tail` prints the last lines

```
$ tail primers.fa
```

Redirects: do it there, not here!

- Many commands write text to the terminal:

```
$ head fileA
```

- Maybe the output is:
  - too long and/or
  - you want to save it on disk

```
$ head fileA > headOfFileA
```

- The reverse also works:

```
$ head < inputFile > headOfInputFile
```

“Pipes”

You can also “pipe” the output of a command into another command

- How many are the files in the current directory?

```
$ ls > listOfFiles
$ wc -l listOfFiles
```

- Why bother with a file? Use a pipe:

```
$ ls | wc -l
```

You can really do a lot of stuff with these

Grep and family

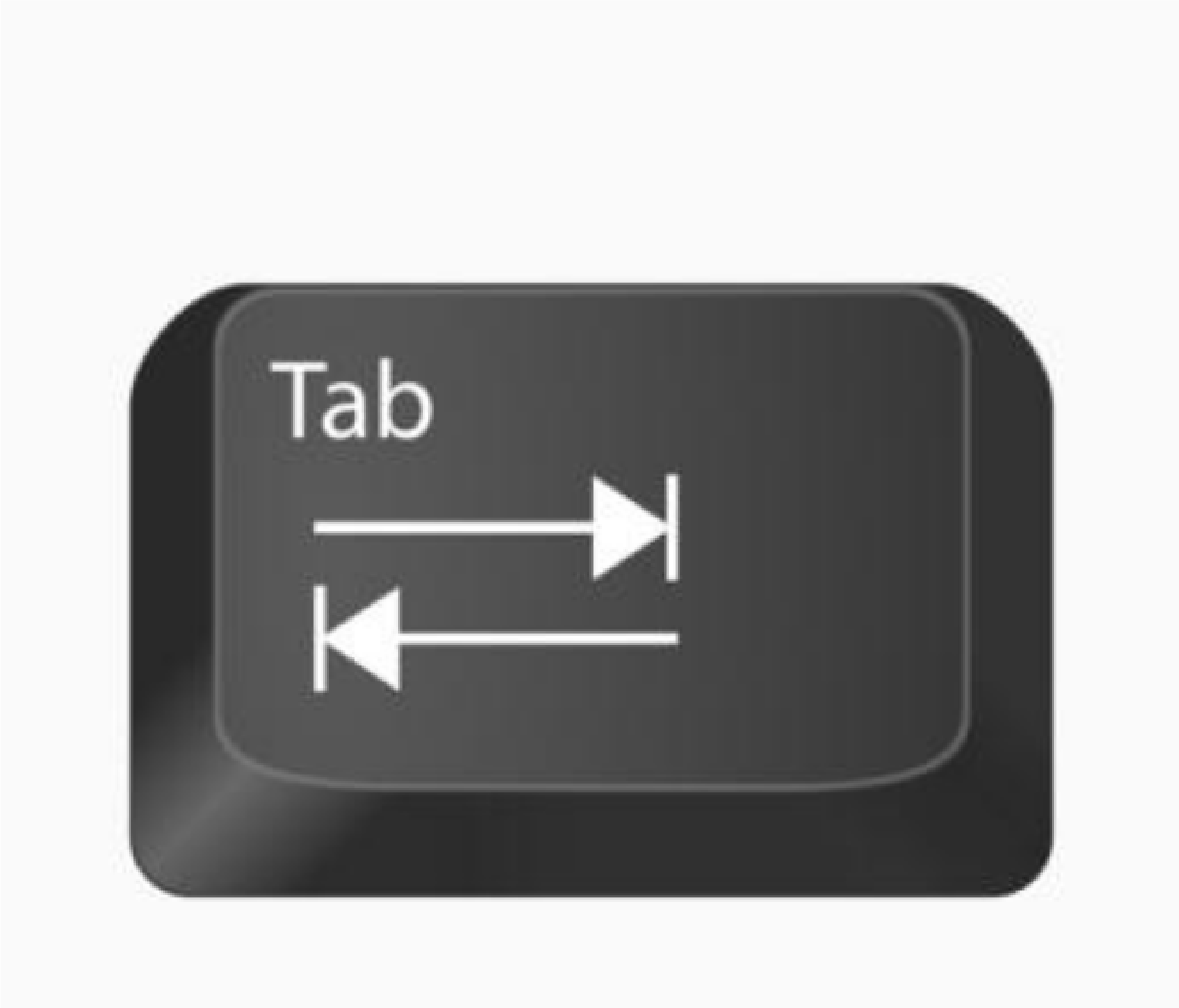
- Often you want to see only some lines in a file:

```
$ grep password secretsOfPippo
password di facebook di Pippo: 123456
password del conto in banca: qwerty
```

- Can be very useful!

- Other useful command for text files:
  - sort, tr, diff, cut, sed, awk, … (there are really many)

Don’t forget…



Why bother?

The textual shell has some disadvantages:

- you need to know the commands (no hints)
- little feedback on what happens
- not very intuitive (or you are not used to it)

So why bother? Because the shell:

- handles big data much better
- GUIs take too long to write