

Open Source compliance: Technical must-knows for legal experts

Open Source Automation Development Lab (OSADL) eG

The terminal Shell command line Directory tree

What is a terminal?

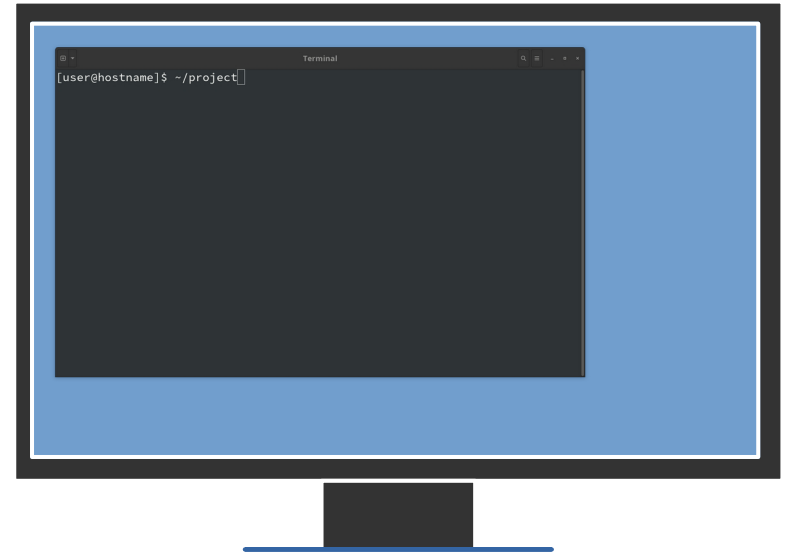
The terminal is the generic interface to enter commands to a computer (=command-line interface or short CLI).

What is a terminal?

A terminal is a software for communicating with the computer via text commands.

Examples:

- Linux: Terminal, XTerm, Konsole
- Apple: MacOS Terminal
- Windows: Git Bash or wsl



Characteristics of a terminal

- The terminal is the generic interface to enter commands to a computer (=command-line interface or short CLI).
- The program that manages text input and program execution is called **shell**.
- In addition, many low-level text-oriented programs are best executed from command line. This includes programs to investigate licensing and module interdependency.

How is the shell used?

After a terminal window is launched or is switched to, the shell is automatically started and prints the so called shell prompt to the terminal such as:

A screenshot of a terminal window. The title bar at the top reads "Terminal". The main area of the terminal is dark gray and contains the text "user@hostname: ~/project\$" in a light gray font, representing the shell prompt.

How is the shell used?

After a terminal window is launched or is switched to, the shell is automatically started and prints the so called shell prompt to the terminal such as:



The image shows a terminal window titled "Terminal". The prompt displayed is "user@hostname:~/project\$". A green L-shaped line highlights the "user" part of the prompt. A green arrow points from the "user" part to a black box containing the text "User name".

```
Terminal
user@hostname:~/project$
```

How is the shell used?

After a terminal window is launched or is switched to, the shell is automatically started and prints the so called shell prompt to the terminal such as:

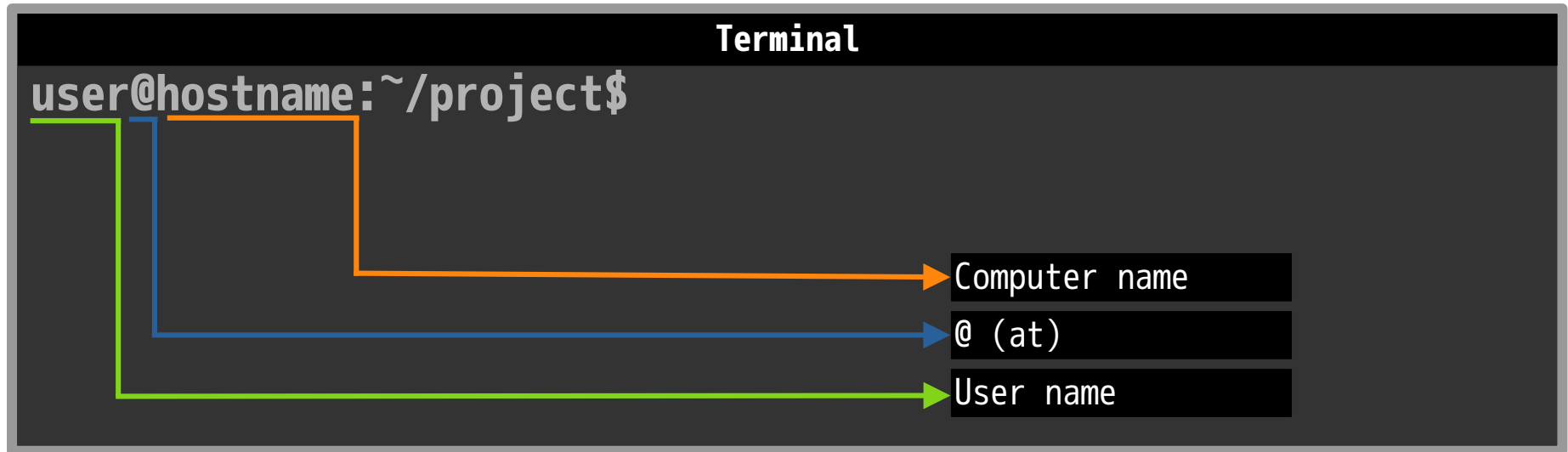


The image shows a terminal window titled "Terminal" with a dark background. The prompt "user@hostname:~/project\$" is displayed in white text. A blue arrow points from the "@" symbol to a callout box containing "@ (at)". A green arrow points from the "user" part of the prompt to another callout box containing "User name".

```
Terminal
user@hostname:~/project$
```

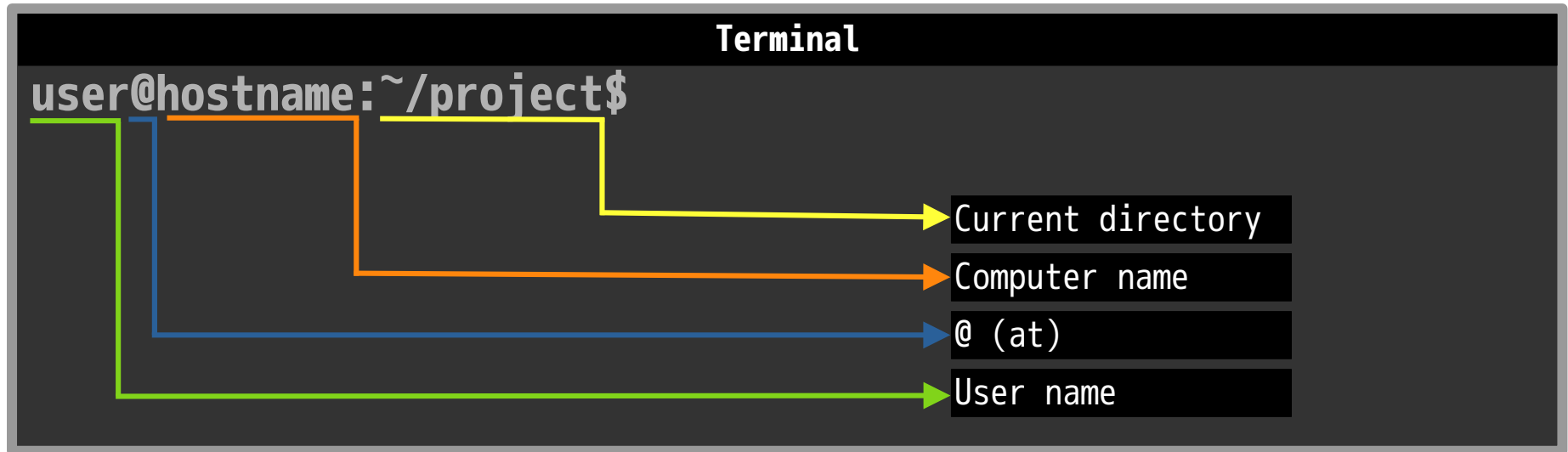

How is the shell used?

After a terminal window is launched or is switched to, the shell is automatically started and prints the so called shell prompt to the terminal such as:



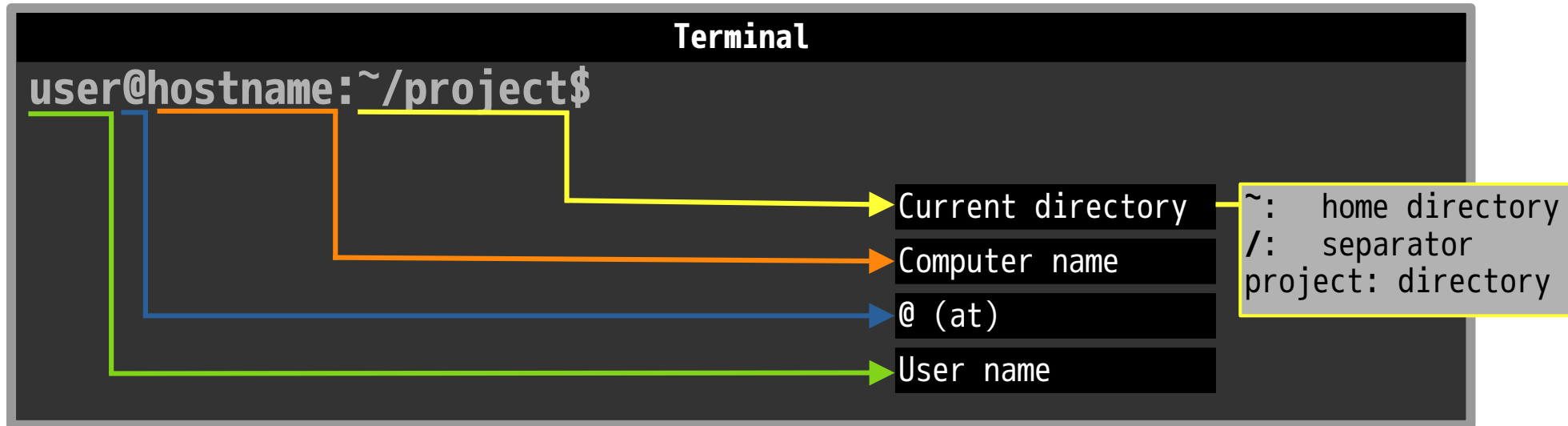
How is the shell used?

After a terminal window is launched or is switched to, the shell is automatically started and prints the so called shell prompt to the terminal such as:



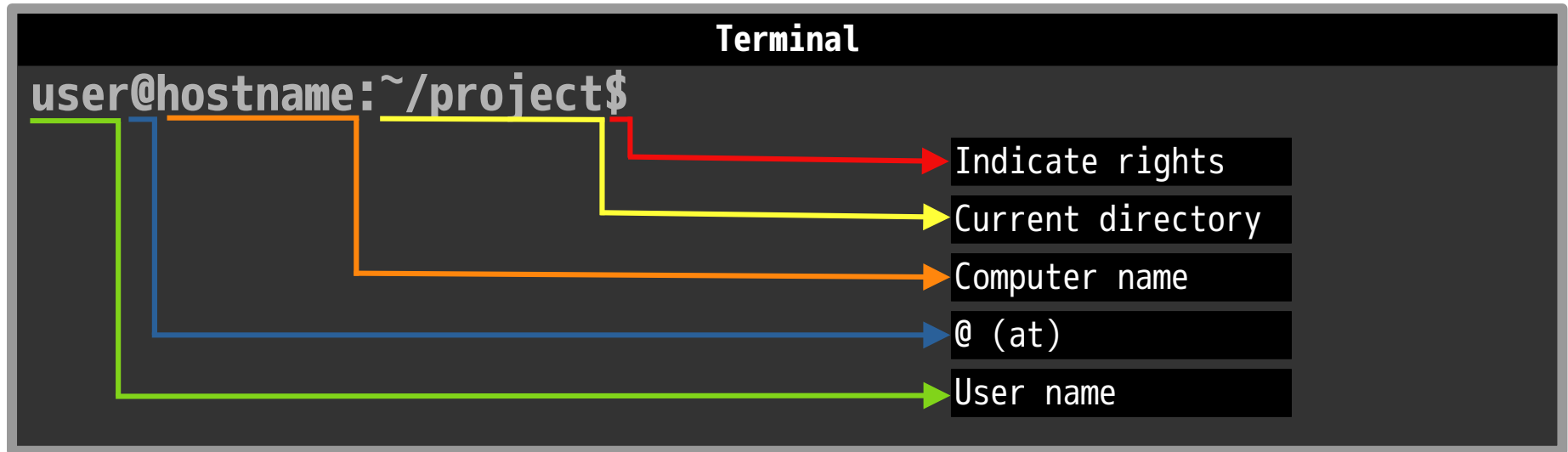
How is the shell used?

After a terminal window is launched or is switched to, the shell is automatically started and prints the so called shell prompt to the terminal such as:



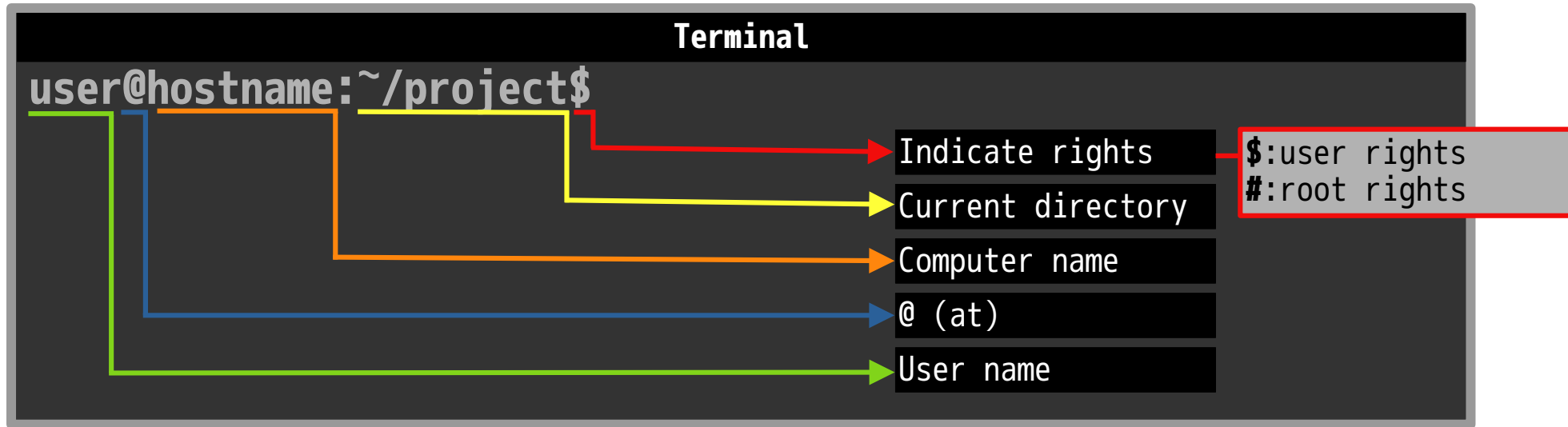
How is the shell used?

After a terminal window is launched or is switched to, the shell is automatically started and prints the so called shell prompt to the terminal such as:



How is the shell used?

After a terminal window is launched or is switched to, the shell is automatically started and prints the so called shell prompt to the terminal such as:



How is the shell used?

After a terminal window is launched or is switched to, the shell is automatically started and prints the so called shell prompt to the terminal such as:



The image shows a terminal window with a dark background. At the top, the word "Terminal" is written in white. Below it, the shell prompt "user@hostname:~/project\$" is displayed in white. A white arrow points from the end of the prompt to a white-bordered box containing the text: "This 'prompts' the user to type a command and to hit the 'Enter' key to submit it to the operating system."

```
Terminal
user@hostname:~/project$
```

This "prompts" the user to type a command and to hit the "Enter" key to submit it to the operating system.

Command execution in the shell

- **“Built-in” commands** are executed internally by the shell itself, enabling resource-saving and fast execution without the need for external binaries (=programs).
 - Examples: `cd`, `pwd`, `echo`, `exit`, `read`, `help`
- **External commands** are executable files stored in the file system. Whenever a command is entered that is not available as built-in command, the shell tries to find a program with the name of the command and executes it if found.
 - Examples: `ls`, `grep`, `cat`, `cp`, `mkdir`, `mv`, `date`

Syntax of the commandline

The general syntax of a command is as follows:

```
Terminal
user@hostname:~/project$ command [options] [arguments]
```

- **command:** The program or command to be executed (e.g. ls, echo, cd).
- **options:** Additional modifiers that change the behavior of the command
- **arguments:** File-/dirname or data to which the command is applied

Example:

```
user@hostname:~/project$ ls -l /home/user
```

→Displays files and directories in the /home/user directory in long format.

“Built-in” commands

- **cd** Change directory
 - `cd` → change to users home directory
 - `cd [dir]` → change to directory [dir]
 - `cd ..` → change to parent directory
- **pwd** Print current working directory
- **echo** Write arguments to the standard output
 - `echo text` → text
- **exit** Exit the shell
- **read** Read from the standard input and split it into fields
- **help** Display information about builtin commands

External commands

- **ls** List information about the FILES
 - `ls -l` → use long listing format
 - `ls -t` → sort by time, newest first
 - `ls -tr` → sort by time, newest last (reverse sorting)
- **mkdir DIR** Create the directory DIR
- **cp SRC DST** Copy SRC to DST
- **mv SRC DST** Rename SRC to DST
- **cat FILE** Print the content of the file FILE
- **grep PTN FILE** Searches for patterns PTN in each FILE
- **man CMD** Display manpage of CMD

File redirection, execution separators and piping

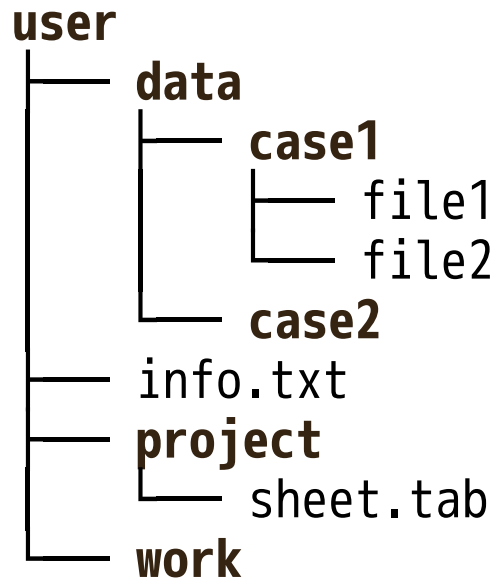
- The output of a command may be redirected to a file using one of the following redirection symbols:
 - **>file** Send output to *file*, create new or overwrite it
 - **>>file** Append output to *file*, create new if it does not exist
- Several commands can be entered in one command line:
 - separated by **;** to indicate sequential execution
 - or by **&** to indicate parallel execution
- A pipe (**|**) can be used to connect commands in Bash
 - `command1 | command2`
 - The output of a `command1` is used directly as input for `command2`

Navigating in the directory tree

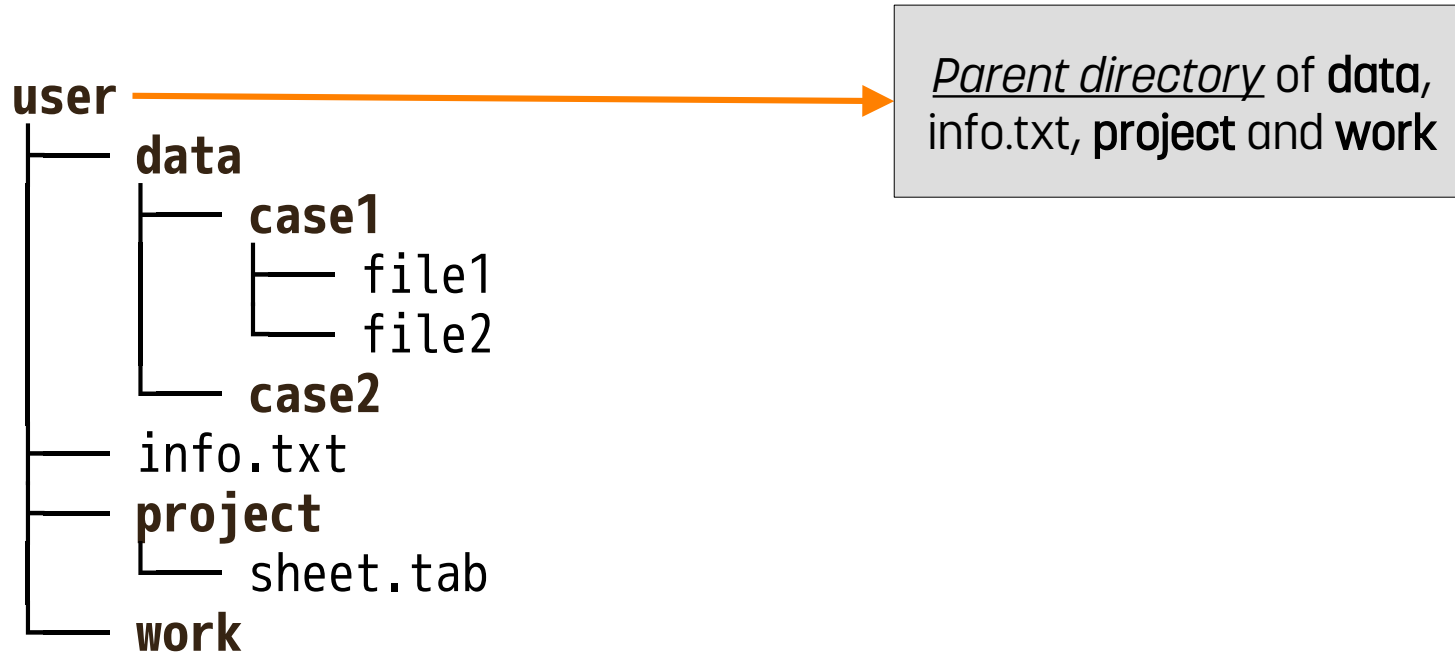
- Files and directories on a storage medium are organized in a tree-like hierarchy:

Navigating in the directory tree

- Files and directories on a storage medium are organized in a tree-like hierarchy:



Navigating in the directory tree



Navigating in the directory tree

