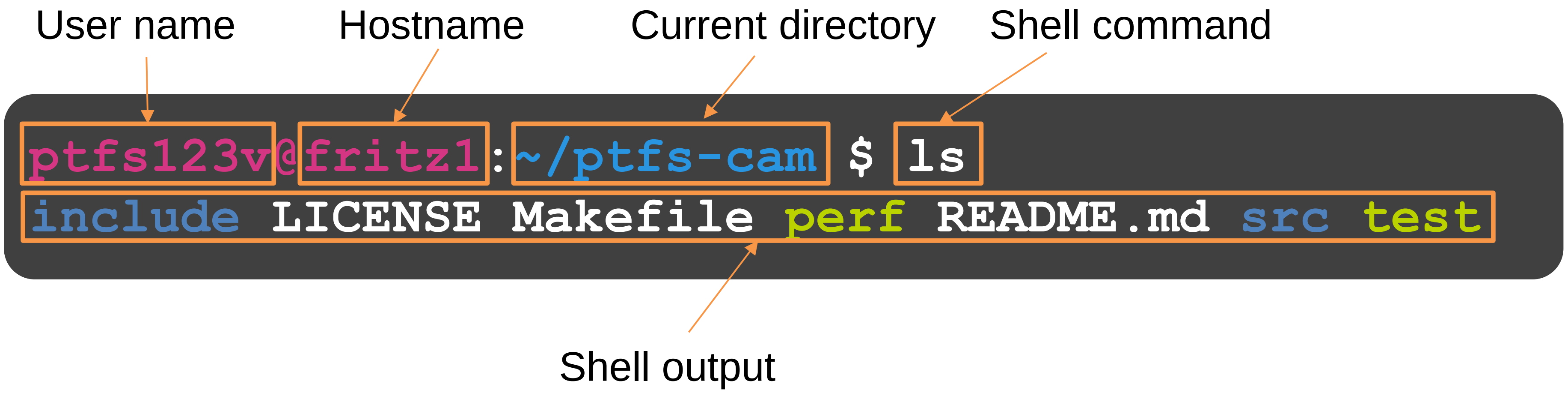
PTfS-CAM

Shell basics

Shell

What is a shell?

- Text oriented (command line) mediator between user and system.
- Waits for commands, interprets them and starts programs.
- Supervises and controls program execution (e.g., scripts).



```
ptfs123v@fritz1:~/ptfs-cam $ ls
include LICENSE Makefile perf README.md src test
```

Shell commands – part 1

Some **basic** shell commands you should know

- **`echo`** – prints the string it is being passed
- **`ls`** – list all files and folders
- **`cd`** – change directory
- **`cp`** – copy files and folders (use **`-r`** for folders)
- **`mv`** – move files and folders
- **`mkdir`** – making a new directory
- **`rm`** – remove files and folders (use **`-r`** for folders)
- **`ssh`** – connect and execute commands on remote host
- **`scp`** – copy files to or from remote host (use **`-r`** for folders)
- **`man`** – print manual or get help fo

There are many many commands available.
Google is your friend.

Environment variables

- Name values pairs in a program’s environment.
- They can control the program’s behaviour.

```
ptfs123v@fritz1:~/ptfs-cam $ echo $HOME
/home/hpc/ptfs/ptf123h
```

- See defined environment variables using `env` shell command.
- To set environment variables use:
  - **`export VARNAME=<value>`**
  - **`setenv VARNAME <value>`** (on csh shell)
- Important characters with special meaning:

  `~` → HOME expansion **`*`** → wildcard **`./`** → current directory

Scripts

- Shell scripts are a sequence of commands written to a file.
- Using scripts allows you to automatize tasks.

Example file **`script.sh`**:

```
#!/bin/bash
echo "Your home is ${HOME}"
echo "Your 2nd argument is $2 and your 1st argument is $1"
```

- Now convert the **`script.sh`** file to an executable using **`chmod`** command.

```
$ chmod u+x script.sh
```

- Run the script. Pass in arguments if required.

```
$ ./script.sh abc 1
```

Scripts

- Use scripts wisely.
- Use scripts to run and collect results. This avoids human errors and documents your run settings.
- There are control flow loops/statements like **`for, while, if`**
- **Always validate the correctness of your script.**
- Output redirection (**>**, **>>**, and **`tee`**) can be handy.
- You can use pipe operator (**|**) to pass the output of one command to another command.
- Some of my favorite shell commands for string parsing : **`grep, cat, cut, find, head, tail, bc , awk ...`**

Shell commands – part 2

- **`vim`** – text editor (requires tutorial for its own)
- **`grep`** – print lines matching a pattern (use **`-R`** for reverse search)
- **`cat`** – concatenate files and print on standard output
- **`cut`** – print selected parts of lines
- **`find`** – find file in directory
- **`head / tail`** – print the N first / last lines of file/output
- **`bc`** – calculator for basic arithmetic
- **`awk`** – pattern scanning and processing language

Hands on: running and collecting results

- Copy the program (square) from `/home/hpc/ptfs/ptfs100h/shell/square` to your home
- The program squares a number and returns the results.

```
ptfs123v@fritz1:~/ptfs-cam $ ./square 4
Lots of
lines
of text
Square = 16
End
```

- Now use a shell script to collect the squares for 1…10 and report the result in a csv file format such that:

```
#val, square
   1, 1
   2, 4
   3, 9
…
```

Hands on: Solution

```
#!/bin/bash
echo "#val, square" > out.csv
for (( i=1; i<=10; i++ )); do
    ./square $i > tmp.txt
    square=$(cat tmp.txt | grep "Square =" | cut -d"=" -f 2)
    echo "$i,${square}" >> out.csv
done
rm tmp.txt
```

Shell commands – part 3

- **`sed`** – stream editor for filtering and transforming text
- **`less`** – view the content of a file/output
- **`pwd`** – print current directory path
- **`htop`** – interactive process viewer

- **`<Ctrl+c>`** – close process
- **`<Tab>`** – auto-complete commands
- **`<Up> / <Down>`** – go element-wise through history
- **`<Ctrl+r>`** – reverse search of history

Useful links

- bash cheatsheet
- text editors: vim tutorial or nano tutorial
- terminal multiplexer: tmux or screen