Command-line Interface (CLI)

Learning objectives

- Understand and explain what a `command-line interface (CLI)` is
- Use basic `commands` in CLI to create and manipulate files and folders
- Supply `params` and `options` to command-line programs

Interfaces

- An `interface` is a way to communicate with a computer
- A `user interface` is generally a visual way to interact with a computer, like a `GUI` or `graphical user interface`
- There is also a concept of `API` or `application program interface`, a way that programs within a computer interact with each other

The first user experience

- The `CLI` or `command-line interface` was one of the first ways a user could communicate with a computer and remains a powerful way today to issue commands to a computer
- Back in the days of mainframes, there was really only one way to interact with the computer, and that was through text

The first user experience (cont.)

- Each OS (Windows, Mac, Linux/Unix) has its own flavor of command-line
- Some examples include `cmd.exe`, `Cygwin`, `PowerShell`, `Terminal`, `iTerm 2`, and `Konsole`

Interactive shell

- The general pattern is that there is a `prompt`, which you might recognize as `C:>` or `$`
- At the prompt, the user can enter a command. A command to show what files are present in the current directory looks like `C:> dir` or `$ ls`

Interactive shell (cont.)

- A user can supply `params` (short for parameters) which tell the program a little more information: `C:> dir C:/Windows` or `$ ls /etc`
- A user can also supply `options`, which are flags or switches, that modify how the command operates: `C:> dir /p` or `$ ls -g`

Other examples of interactive shells

- Some programming languages have interactive shells, including Python, Ruby, and JavaScript
- These allow you to enter commands that are immediately interpreted on the fly (as opposed to compiled languages like C++ or Java)

Let's learn some commands

- Navigating: `ls`, `pwd`, `cd`
- File editing: `touch`, `nano`, `vim`
- File manipulation: `cp`, `mv`, `rm`, `mkdir`
- Viewing files: `cat`, `less`, `more`, `tail`
- Downloading files: `curl`, `wget`
- Understanding commands: `man`, `-h` flag

Reference:

`http://cli.learncodethehardway.org/bash_cheat_sheet.pdf`

Navigating files and folders

- `ls`
- `pwd`
- `cd`

File editing

- `touch`
- `nano`
- `vim`

File manipulation

- `cp`
- `mv`
- `rm`
- `mkdir`

Viewing files

- `cat`
- `less`
- `more`
- `tail`

Downloading files

- `curl`
- `wget`

Understanding commands

- `man`
- `-h` flag

File permissions

```
$ ls -l
total 40
-rwxrw-r--   1 john  staff
-rwxrw-r--   1 john  staff
-rwxrw-r--   1 john  staff
drwxr-xr-x   5 john  staff
```

File permissions

- The 10 characters in the beginning are a `bitmask` that represents the permissions of a file
- The 1st is `d` or `-`, where "d" means directory and "-" means it's not a directory, i.e. it's a file
- The remaining 9 positions are flags for `read`, `write`, or `execute` among reiterated 3 times for `user`, `group`, and `other`
- Here `john` and `staff` are the user and group that the file belongs to

Advanced topics

- Changing permissions: `chmod`, `chown`, `chgrp`
- Regular expressions: `grep`
- Redirections: `echo "hello world" >> test.txt`
- Pipes
  - `ls | sort -r`
  - `grep "light" * | more`

In-class activity (ICA)

Create a file in CLI named `firstname-lastname.txt` with the following information in it:

```
First Name: Tiffany
Last Name: Pang
Birthdate: 02/01/1990
Address: 4129 Judah St. San Francisco
Phone Number: 415.272.4362

Emergency Contact: John Cadengo
Emergency Contact Phone Number: 909.84
```

Summary

- The command-line interface is a powerful way to interact with the computer
- It has a long, rich history in computing from the days of mainframes, and has since splintered off in various ways
- It will be a primary tool in your toolkit as a developer

Reference:

`http://cli.learncodethehardway.org/bash_cheat_sheet.pdf`