

Environment

What We'll Cover

- **Computer organisation: files & directories**
 - tree architecture
- **Terminal?**
 - Shells, Prompts & Commands
- **Developer-friendly naming & folder rules**
- **Machine navigation**
 - GUI vs. Terminal (CLI)
- **Storage**
 - Clouds storage backups
- **Live Demo Walkthrough + Q&A**

Computer organisation

Files and directories

What is a file & file extension?

- **File**

- A file is a raw block of data stored with a name and a *file extension*
e.g., `app.py`, `index.html`, `data.json`

- **File extension**

- Tells Operating System and Code Editors (like VS Code) how to parse and highlight syntax

- **Show File Extensions –**

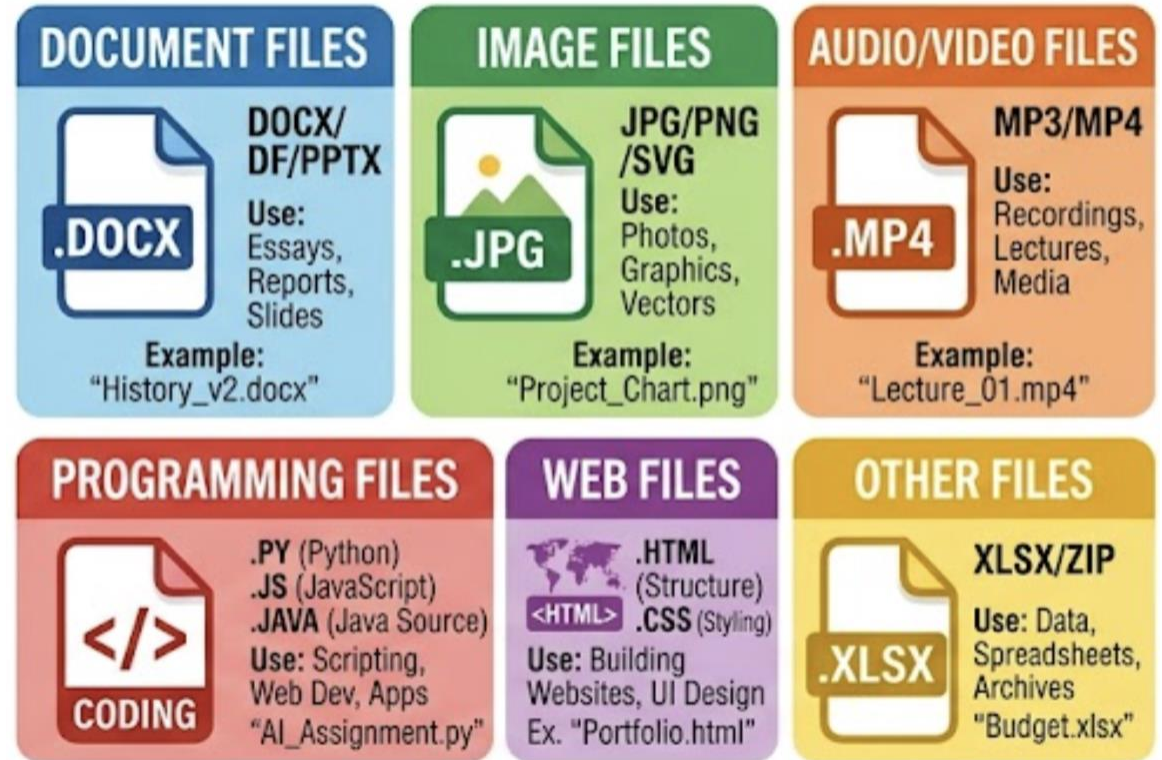
- **Windows:**

File Explorer → Click View → Check File name extensions

- **Show File Extensions –**

- **macOS:**

Finder → Settings → Advanced → Check Show all filename extensions



'Root', 'Home' & 'Paths'

- **Root & user home**

- Computers arrange files as inverted tree starting at Root
- Windows Root: `C:\`
- macOS Root: `/`
- Personal **Home** is the top-level directory
 - Windows: `C:\Users\YourName`
 - macOS: `/Users/YourName (~)`
- Your stuff goes in here

- **Absolute paths are full address from root**

Windows: `C:\Users\alex\degree\year-1\tri-a\p0\code\week-01\app.py`
macOS: `/Users/alex/degree/year-1/tri-a/p0/code/week-01/app.py`

- **Relative Path: navigation relative to current location, e.g:**

Navigate down: `/python/app.py` or
Navigate up: `../data.csv`

Try It Yourself #1

- **Step 1: Navigate to Home**

- Open your File Explorer (Windows) or Finder (macOS) and navigate to your Home folder

- **Step 2: Identify Paths & Root**

- What's your current absolute path? What is the designation for Root on your operating system (C: \ vs /)?

- **Bonus Challenge**

- Locate any file in your folder structure and note its full extension. Verify if extensions are hidden.

Terminal

Terminal

- **What Is It?**

- The terminal (aka 'shell', 'command line', 'CLI')
- Text-based way to control your computer
type a command, hit Enter, get a result

- **Where to Find It**

- Windows: Terminal / PowerShell app
- macOS: Terminal app
- In VS Code the 'Terminal' window interfaces to the one on the computer

- **Prompt (> \$ %)**

- where you type the commands
- configurable

Command examples

Mac	PowerShell	Action / Purpose	Example
<code>pwd</code>	"	Print current working location	<code>pwd</code> → <code>/Users/alex</code>
<code>ls</code>	"	List contents of the current folder	<code>ls -a</code> (-a parameter shows hidden files)
<code>cd</code>	"	Change directory (move around)	<code>cd projects/python</code>
<code>mkdir</code>	"	Create a new directory / folder	<code>mkdir python-app</code>
<code>touch</code>	New-Item	Create a new empty file	<code>touch app.py</code>
<code>rm</code>	"	Delete a file permanently — no trash bin!	<code>rm temp.log</code> (use with caution)

Common Errors & What They Mean

- **command not found**
 - The command doesn't exist, or isn't recognized on your system's `PATH`
- **No such file or directory**
 - That path doesn't exist — double-check it with `pwd` and `ls` first
- **Permission denied**
 - You don't have rights to that file or folder.
 - Permissions need changing
- **Read error messages top to bottom**
 - usually tell you exactly what went wrong

Neatness and backup

Files and directories

Backup

File naming

- **Eliminate spaces**

- Terminals parse spaces as parameter separators. “`my file.py`” becomes two separate arguments!
- Use `snake_case` instead – e.g. `my_file.py`

- **Dedicated workspace**

- Never code in Desktop or Downloads!
- Create a clean hierarchy inside your User Home:
`YOUR_HOME/SOME_PATH_PREFIX/projects/python/app.py`

- **Case Conventions**

- `kebab-case` for files/folders or `PascalCase` a.k.a `UpperCamelCase`
- `snake_case` for Python scripts
- ... you will learn others on your degree

Folder naming

- **Eliminate spaces**

- All paths are best closed – i.e. no spaces
- Yes: `C:\Users\alex\degree\year-1\tri-a\p0`
- No: `C:\Users\alex\degree\year 1\trimester A\p0`

- **Short names for folders but should be descriptive**

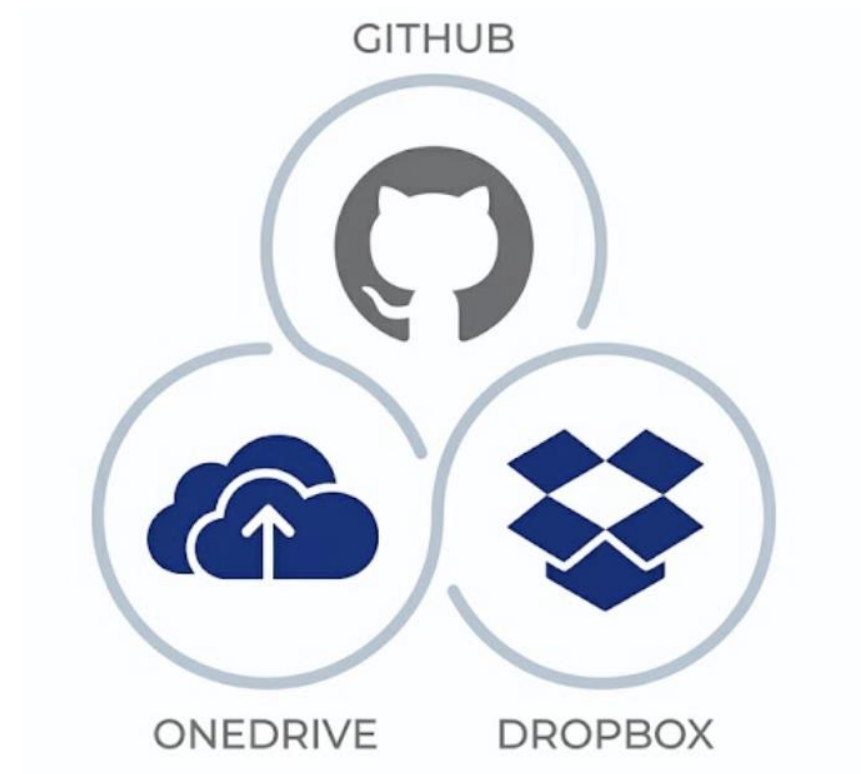
- `p0` rather than `Programming-zero`

- **Dedicated workspace**

- Never code in Desktop or Downloads!
- Create a clean hierarchy inside your User Home:
`YOUR_HOME/SOME_PATH_PREFIX/projects/python/app.py`
- ... and in some cloud account like OneDrive (Dropbox) etc.

Backup (automatic)

- **Use a cloud service**
- **OneDrive**
 - Microsoft Office 365
 - Fine for now for code
- **Version control**
 - Like git
 - Meet later on your degree



Version control

- **Code management**

- system that tracks changes to code over time, allowing you to review project history, collaborate safely with others, and revert to previous working states

- **Cloud based services**

- GitHub: cloud platform 'hosts' remote copies of code
- GitLab, Bitbucket

- **Backups**

- Unlike OneDrive/iCloud, Git only backs up what you explicitly want.
- Can ignore programming 'junk'.

Conclusions

- **Keep computer organized into files & directories**
 - Practice: or forever be inefficient (and possibly lost)
- **Get familiar with your computer Terminal**
 - Practice: another sure path towards efficiency
- **Developer-friendly naming & folder rules**
 - Rules are for good reason. Use them!
- **Cloud Storage and backup**
 - Don't keep your work in a silly place
 - ... where it can be easily deleted and lost

Thanks!