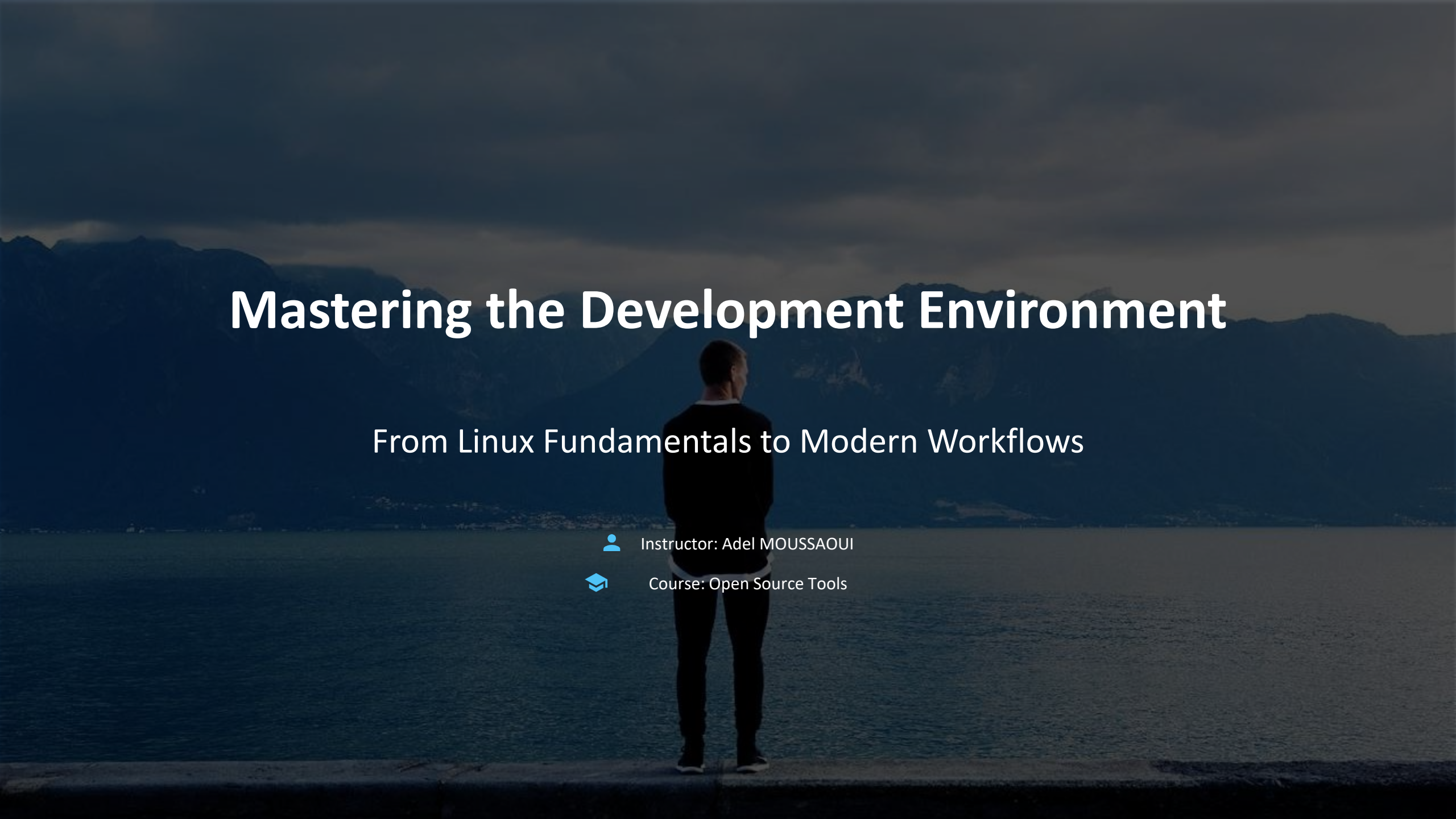


Mastering the Development Environment

From Linux Fundamentals to Modern Workflows

Instructor: Adel MOUSSAOUI

Course: Open Source Tools

What is Linux? More Than an OS

Linux is a family of open-source, Unix-like operating systems.

The Core Components:

The Kernel

The core program that manages hardware

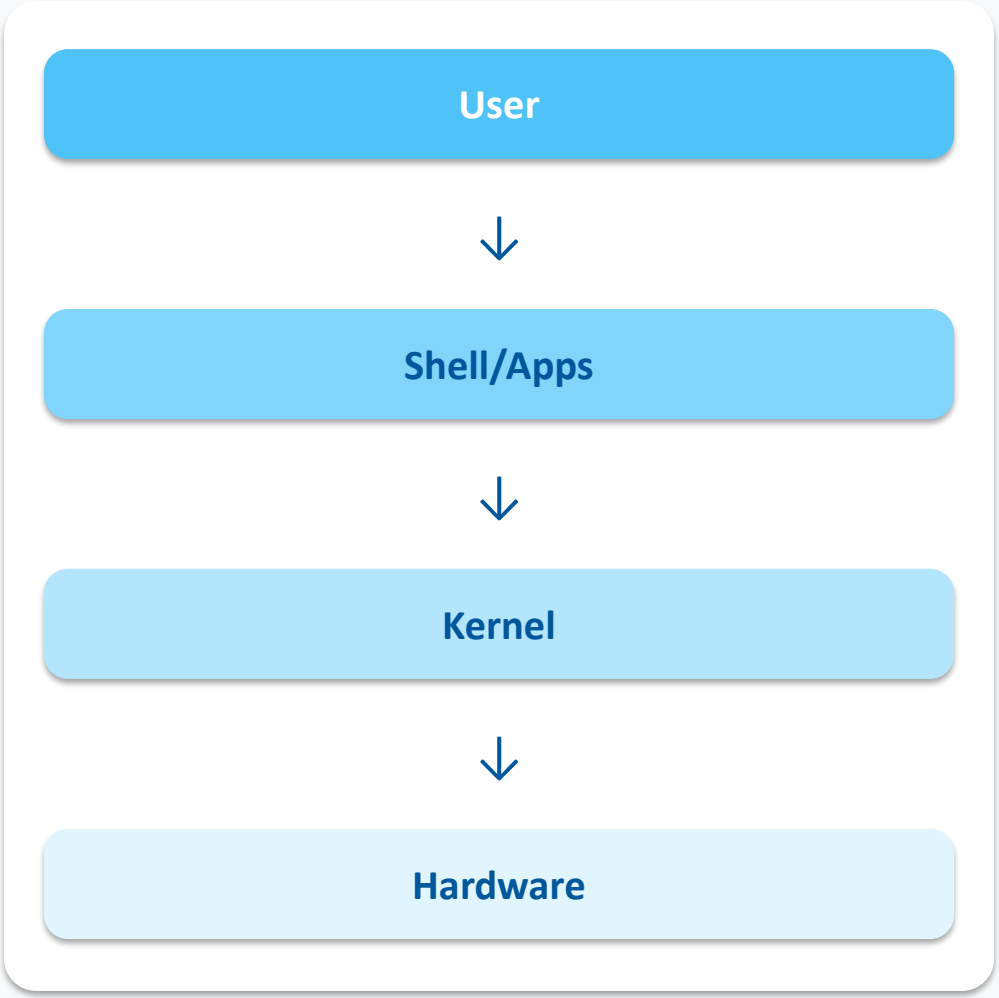
System Tools (GNU)

The essential utilities (shell, compilers, commands,)

The Shell

A command-line interface for interacting with the system

Together, this is often called GNU/Linux



Linux Distributions: Different Flavors

A Distribution (Distro) is a ready-to-install OS package built on Linux.

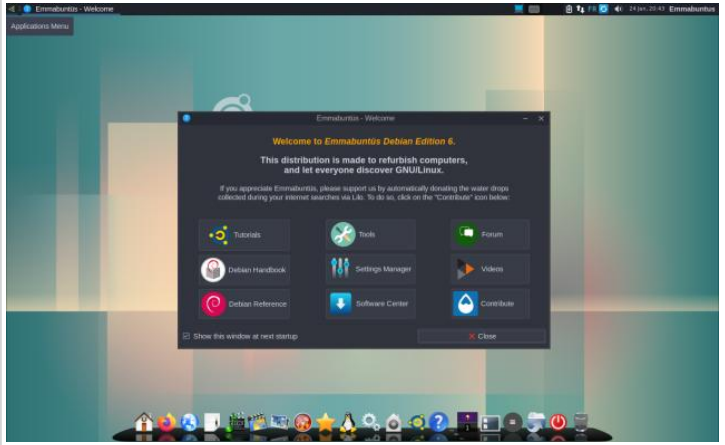
What's in a Distro?

- **The Linux Kernel**
- **A package manager**
  For installing software
- **A desktop environment**
  The GUI
- **Pre-installed applications**

DistroWatch.com

Ubuntu

Great for beginners



Debian

Known for its stability

Why Linux for Development?

Ubiquity
Powers most web servers, cloud infrastructure, and supercomputers

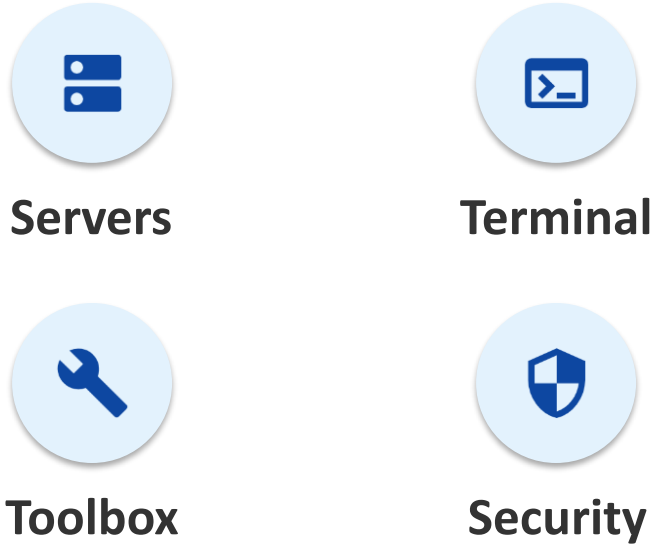
Powerful Command Line
The shell is a developer's superpower for automation and scripting

Native Tooling
Programming languages (Python, C++) and tools (Git, Docker) are designed for Linux first

Package Managers
Install libraries and tools with a single command (e.g., apt, dnf)

Stability & Control
Less bloat, more control over your environment

Development Ecosystem



Getting Hands-On: The Terminal

The Terminal is your text-based window to control the computer.

Key Components:

The Terminal

Text-based window to control the computer

The Shell

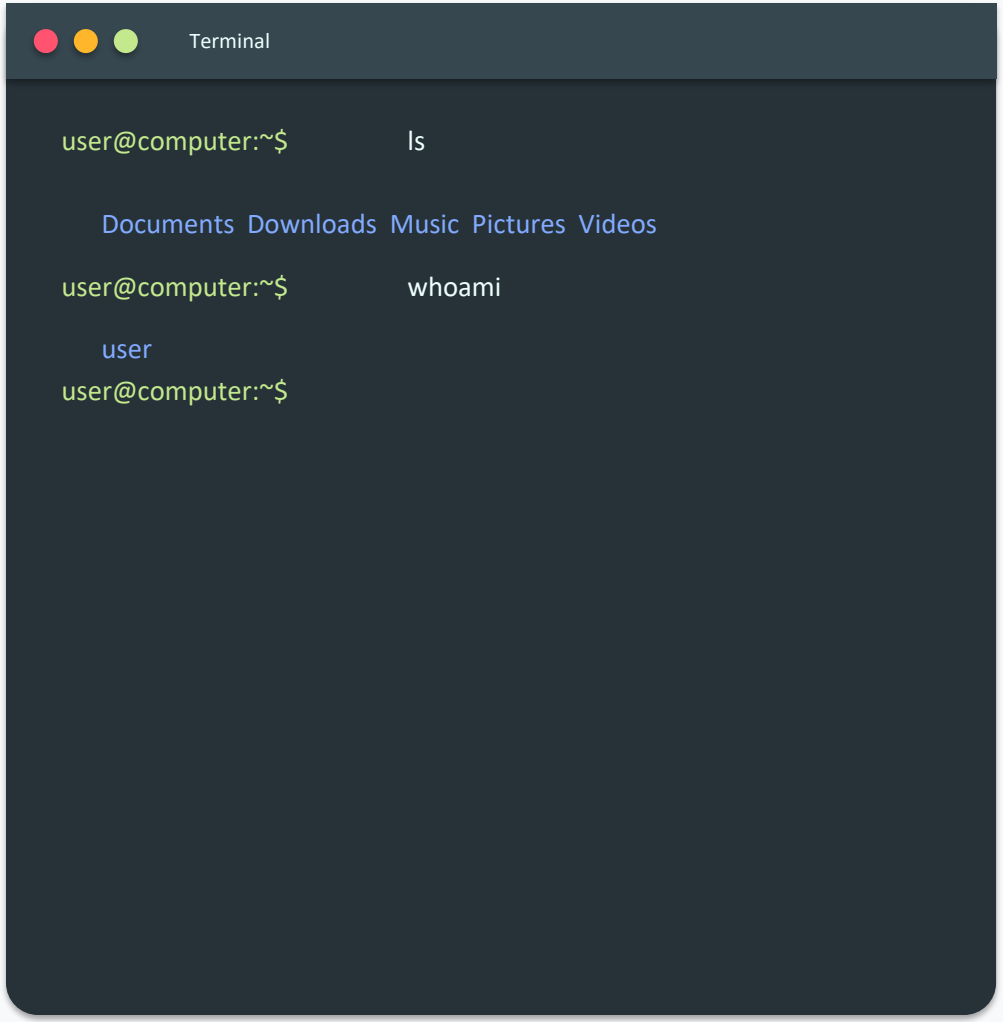
Program inside terminal that interprets commands (Bash, Zsh)

The Prompt

user@computer:~$

```
user @computer : ~$
```

▶ **Your first command: whoami**



The Essential Toolchain

Every developer needs a basic set of tools:

Text Editor

To write code efficiently with syntax highlighting and features

Compiler/Interpreter

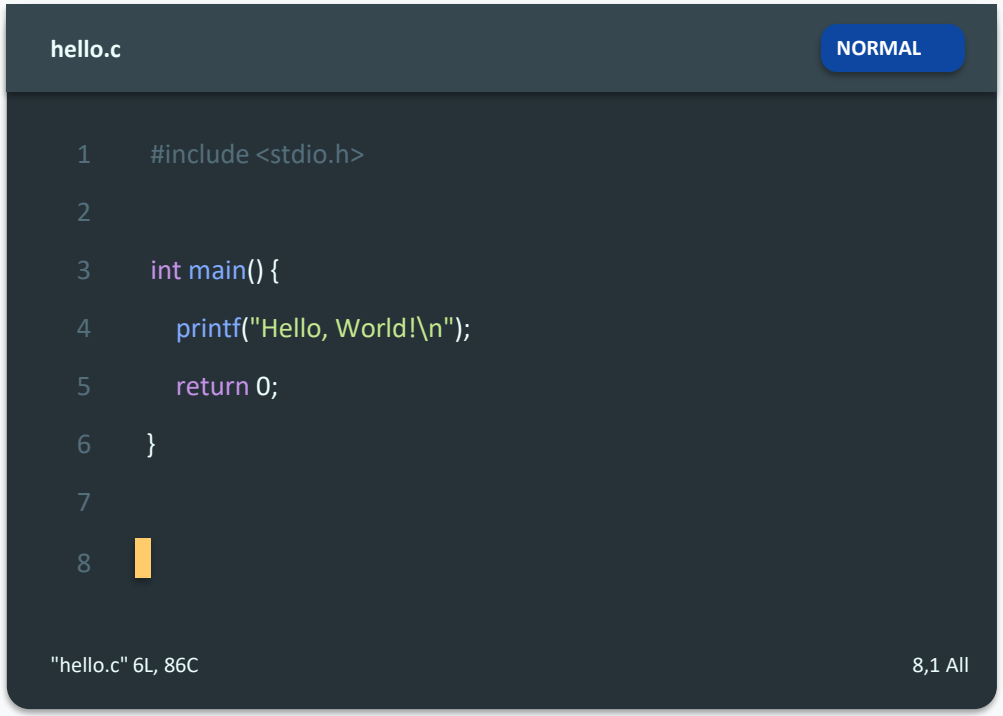
To turn code into a running program

Version Control

To track changes and collaborate effectively

Vim: The Ubiquitous Editor

A powerful, **modal text editor** that runs entirely in the terminal.
Why learn it? It's pre-installed on almost every Linux server and macOS machine.



Modal Editing:

- **Normal Mode**
  For navigation and commands (Press ESC)
- **Insert Mode**
  For typing text (Press i)
- **Command-Line Mode**
  For saving and quitting (Press :)

Survival Commands:

Start typing — **ESC** Back to Normal Mode

:

:wq Save and Quit — **:q!** Quit without saving

Type vimtutor in your terminal for the best interactive guide

Compiling Code with GCC

GCC (GNU Compiler Collection) is the original open-source compiler.

It turns human-readable source code (e.g., C, C++) into machine-executable programs.

The Basic Workflow:

1. **Write code**

   Create **hello.c** with an editor (Vim!)

2. **Compile it**

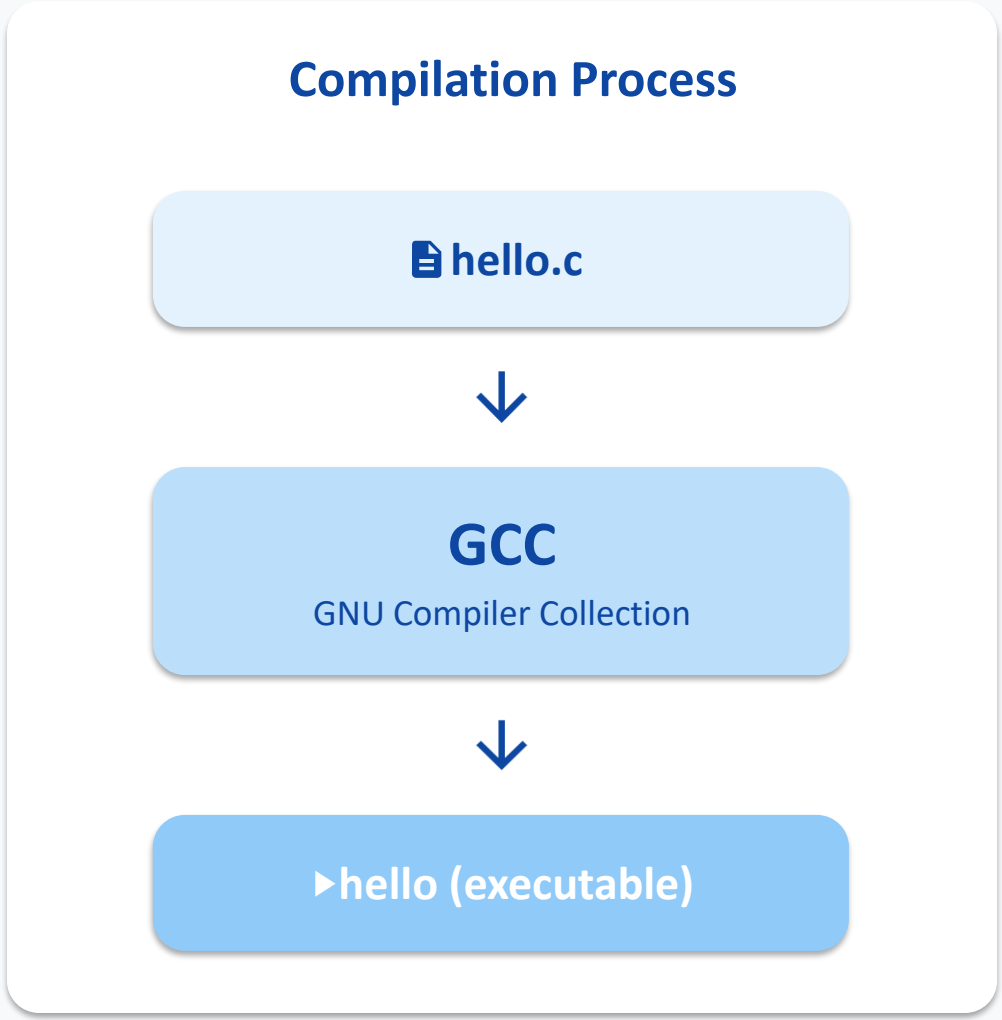
```
gcc -o hello hello.c
```

3. **Run it**

```
./hello
```

> **GCC is used to build the Linux kernel itself and countless other open-source projects**

Compilation Process



The Power of the Package Manager

An **"App Store"** for your command line that automates installing, updating, and removing software and its dependencies.

Examples by Distro:

Ubuntu/Debian

```
sudo apt install git
```

Fedora/RHEL

```
sudo dnf install git
```

Arch/Manjaro

```
sudo pacman -S git
```

Core Commands:

Command Line App Store

update

Refresh the list of available software

install

Install a new package

upgrade

Upgrade all installed packages

Instead of searching the web for software,

you use your package manager

One command to install anything

It handles all the messy details,

like finding and installing required libraries

Essential Command-Line Tools

These tools are the **"swiss army knife"** of the command line.

grep

Searches for text patterns within files

```
grep "error" logfile.txt
```

man

Your built-in manual for commands

```
man ls
```

sudo

"SuperUser Do" - runs commands with administrative privileges

```
sudo apt update
```

chmod

Changes file permissions (read, write, execute)

```
chmod 755 script.sh
```

ssh

Securely connects to a remote server

```
ssh user@server.com
```

Introduction to Build Systems: Make

For complex projects, typing long gcc commands is **inefficient**.

make automates the build process
It reads instructions from a file called Makefile

Simple execution
You simply run **make** in the terminal

Dependency management
Only rebuilds what has changed

Example Makefile for a C project:

```
hello    : hello.c
    gcc -o hello hello.c
```

```
$ make
```

Makefile

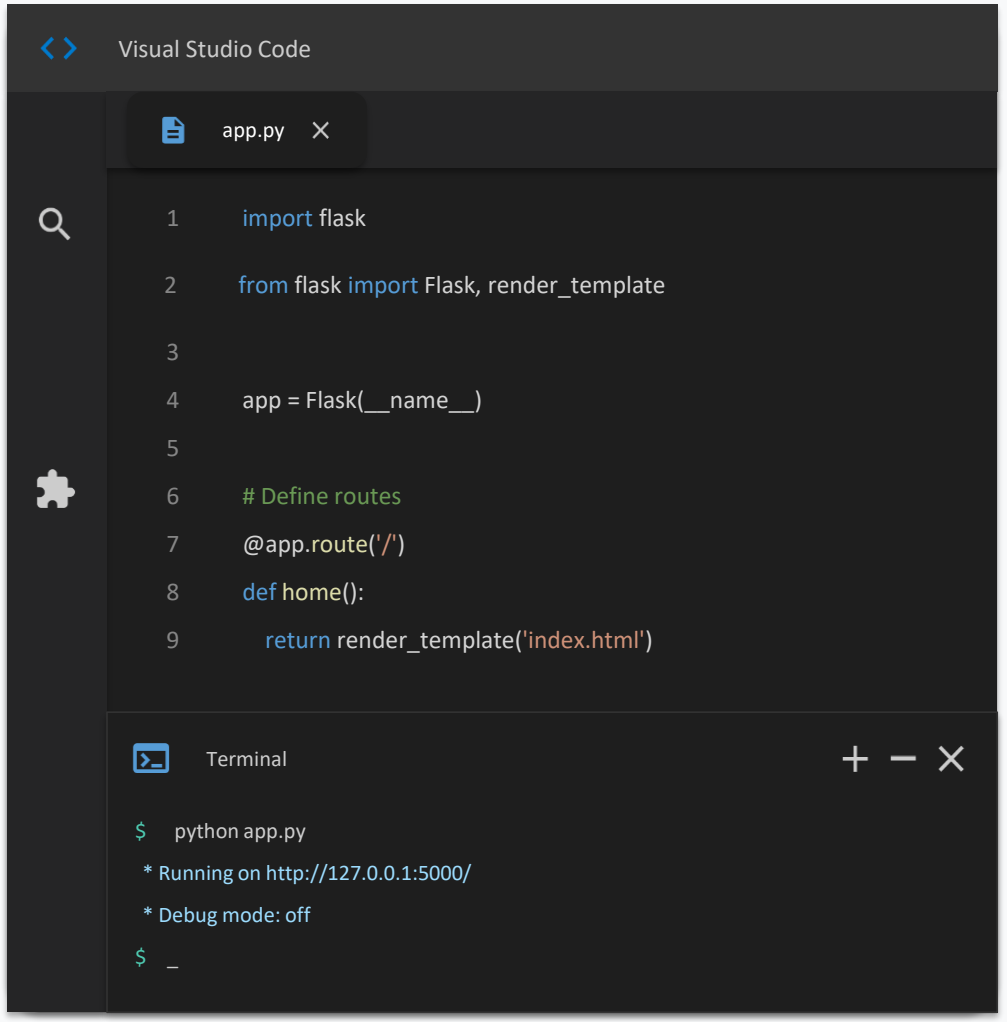
```
# Simple Makefile for hello.c

hello    : hello.c
    gcc -o hello hello.c

clean    :
    rm -f hello

# Usage:
# make - builds the program
# make clean - removes the executable
```

make understands dependencies. If hello.c changes, make will recompile it. If it hasn't changed, it does nothing.

The Modern Code Editor: VS Code

A **free, open-source, cross-platform** editor from Microsoft.

Strikes a perfect balance between a simple text editor and a full-blown IDE.



Key Features:

Syntax Highlighting & IntelliSense
Smart code completion and error detection

Integrated Debugging
Set breakpoints, step through code, inspect variables

Integrated Terminal
Run shell commands without leaving the editor

Extensible Marketplace
Add languages, themes, and tools via extensions

code.visualstudio.com

The Integrated Development Environment (IDE)

An IDE is an **all-in-one suite** for software development that includes code editor, compiler, debugger, and version control, all tightly integrated.

IDE vs. Code Editors

IDEs

Heavier, more features, often language-specific

Code Editors (VS Code)

Lighter, faster, highly customizable via extensions

The line between a powerful editor like VS Code and a lightweight IDE is blurring

Popular Open-Source IDEs

Eclipse

Primarily for Java development

PyCharm Community Edition

Specialized for Python development

IntelliJ IDEA Community

Supports multiple JVM languages

Version Control with Git: The Basics

Git is the most widely used **modern version control system**.

It tracks changes to your code over time, allowing you to revert files or entire projects to a previous state.

GitHub GitLab Bitbucket

Git SCM

Basic Workflow:

1. **Create a new repository**
   `git init`
2. **Stage your changes**
   `git add`
3. **Save a snapshot of your changes**
   `git commit -m "message"`

Git Repository Structure

Local Repository

On your machine

Remote Repository

On GitHub/GitLab

Recap: Your Developer Toolkit

The Foundation

- Linux OS

The Classics

- Vim
- GCC
- Make

The Modern Powerhouse

- VS Code
- Git

Q&A & Resources

Thank You!

Resources for Your Journey

Linux Command Line
Linux Journey

Vim
vimtutor (in your terminal)

Git
Pro Git Book (Free!)

VS Code
Official Documentation

Questions?