Welcome to this CoGrammar Lecture:

The Terminal & Version Control

The session will start shortly...

Questions? Drop them in the chat.
We'll have dedicated moderators
answering questions.

Software Engineering Session Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(Fundamental British Values: Mutual Respect and Tolerance)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** throughout this session, should you wish to ask any follow-up questions.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: **Questions**

Software Engineering Session Housekeeping cont.

- For all **non-academic questions**, please submit a query: **www.hyperiondev.com/support**
- Report a **safeguarding** incident: **www.hyperiondev.com/safeguardreporting**
- We would love your **feedback** on lectures: **Feedback on Lectures**

Enhancing Accessibility: Activate Browser Captions

Why Enable Browser Captions?

- Captions provide **real-time text for spoken content**, ensuring inclusivity.
- Ideal for individuals in noisy or quiet environments or for those with **hearing impairments**.

How to Activate Captions:

1. **YouTube or Video Players:**
   - Look for the CC (Closed Captions) icon and click to enable.
2. **Browser Settings:**
   - Google Chrome: Go to *Settings* > *Accessibility* > *Live Captions* and toggle ON.
   - Edge: Enable captions in *Settings* > *Accessibility*.

Safeguarding & Welfare

We are committed to all our students and staff feeling safe and happy; we want to make sure there is always someone you can turn to if you are worried about anything.

If you are feeling upset or unsafe, are worried about a friend, student or family member, or you feel like something isn't right, speak to our safeguarding team:

Ian Wyles
Designated Safeguarding Lead

Simone Botes

Nurhaan Snyman

Rafiq Manan

Ronald Munodawafa

Tevin Pitts

Scan to report a safeguarding concern

or email the Designated Safeguarding Lead:
Ian Wyles
safeguarding@hyperiondev.com

CoGrammar HyperionDev

Stay Safe Series:

Mastering Online Safety One Week or Step at a Time

While the digital world can be a wonderful place to make education and learning accessible to all, it is unfortunately also a space where harmful threats like online radicalisation, extremist propaganda, phishing scams, online blackmail and hackers can flourish.

As a component of this BootCamp the *Stay Safe Series* is designed to guide you through essential measures in order to protect yourself & your community from online dangers, whether they target your privacy, personal information or even attempt to manipulate your beliefs.

Trustworthy Websites: How to Spot Secure Sites

- Look for the padlock.
- Check if there is a valid SSL/TLS certificate.
- **Look for a site seal.**
- **Check if the URL is legitimate.**
- Pop-up and Redirection ads are a red flag.

Skills Bootcamp Progression Overview

✅ Criterion 1 - Initial Requirements

Specific achievements within the first two weeks of the program.

To meet this criterion, students need to, by no later than **01 December 2024:**

- **Guided Learning Hours** (GLH)**:** Attend a minimum of 7-8 GLH per week (lectures, workshops, or mentor calls) for a total minimum of **15 GLH**.
- **Task Completion:** Successfully complete the **first 4 of the assigned tasks**.

✅ Criterion 2 - Mid-Course Progress

Progress through the successful completion of tasks within the first half of the program.

To meet this criterion, students should, by no later than **12 January 2025:**

- **Guided Learning Hours** (GLH)**:** Complete at least **60 GLH**.
- **Task Completion :** Successfully complete the **first 13 of the assigned tasks**.

CoGrammar

Skills Bootcamp Progression Overview

✅ Criterion 3 – End-Course Progress

Showcasing students' progress nearing the completion of the course.

To meet this criterion, students should:

- **Guided Learning Hours** (GLH)**:** Complete the **total minimum required GLH,** by the **support end date.**
- **Task Completion : Complete all mandatory tasks**, including any necessary resubmissions, by the end of the bootcamp, **09 March 2025.**

✅ Criterion 4 - Employability

Demonstrating progress to find employment.

To meet this criterion, students should:

- **Record an Interview Invite:** Students are required to record proof of invitation to an interview by **30 March 2025**.
  - **South Holland Students** are required to proof and interview by **17 March 2025.**
- **Record a Final Job Outcome :** Within 12 weeks post-graduation, students are required to record a job outcome.

CoGrammar


SKILLS FOR LIFE
SKILLS BOOTCAMPS
Department for Education
CoGrammar
The Terminal & Version Control

Learning Objectives & Outcomes

- Grasp a basic knowledge of the Terminal
- Identify the basic concepts of version control and Git.
- Explain the purpose and benefits of version control systems.
- Describe the basic commands and operations in Git.
- Initialise a Git repository.
- Stage and commit changes to a repository.

Version Control: The "Time Machine" for Code

CoGrammar

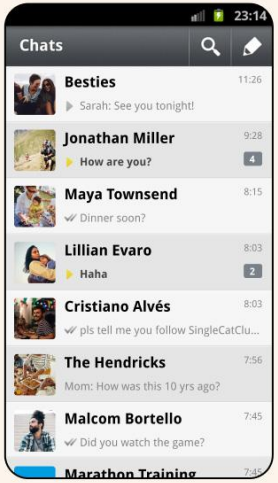
Relevance

- Ever wondered how apps like Instagram and WhatsApp keep getting better without breaking?
- How do developers manage to add new features and fix bugs without chaos?
- The answer lies in a powerful tool called **Version Control**.

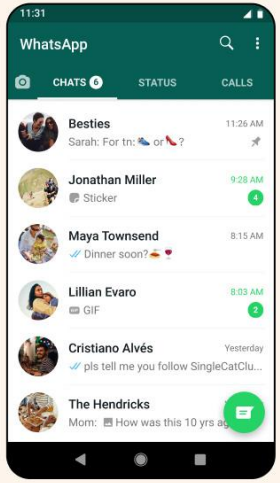
Relevance

Design over the years

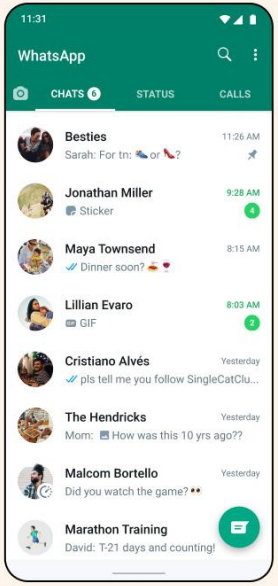
2011



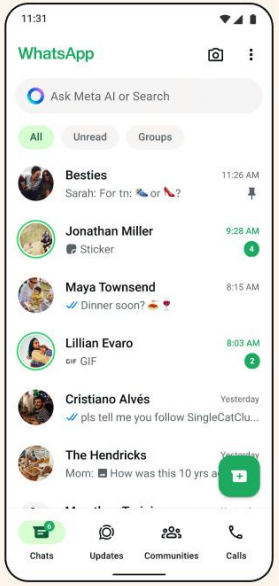
2012



2020



2021



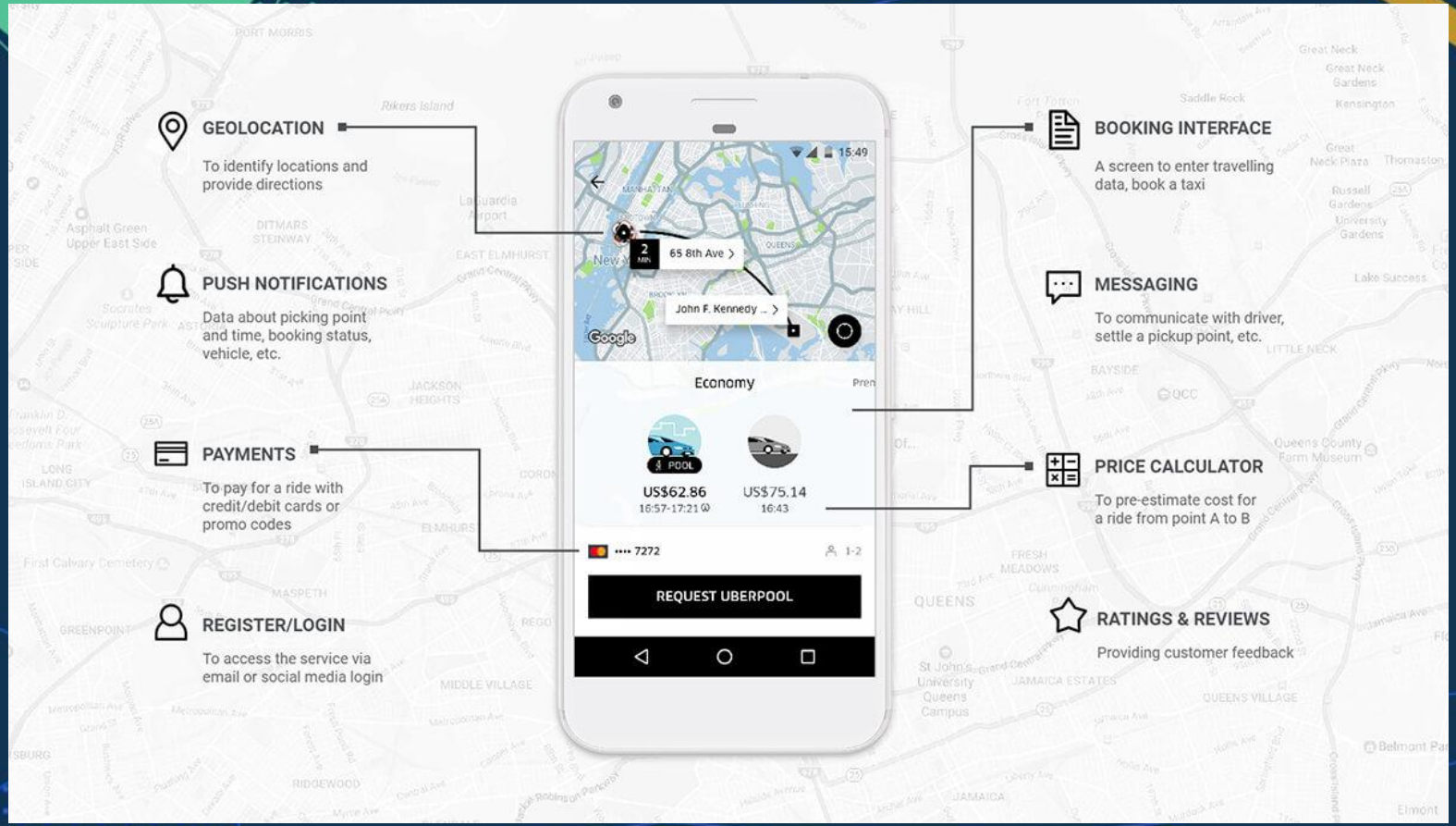
2024

CoGrammar

Real-World Example

- Each update you see has a team of developers behind it, each contributing their part. But how do they track who changed what? And if something goes wrong, how do they roll back to a previous version?

Relevance



CoGrammar

Introduction to Version Control

- Just as you can track changes in a **Google Doc**, developers use Version Control to manage and track code changes.

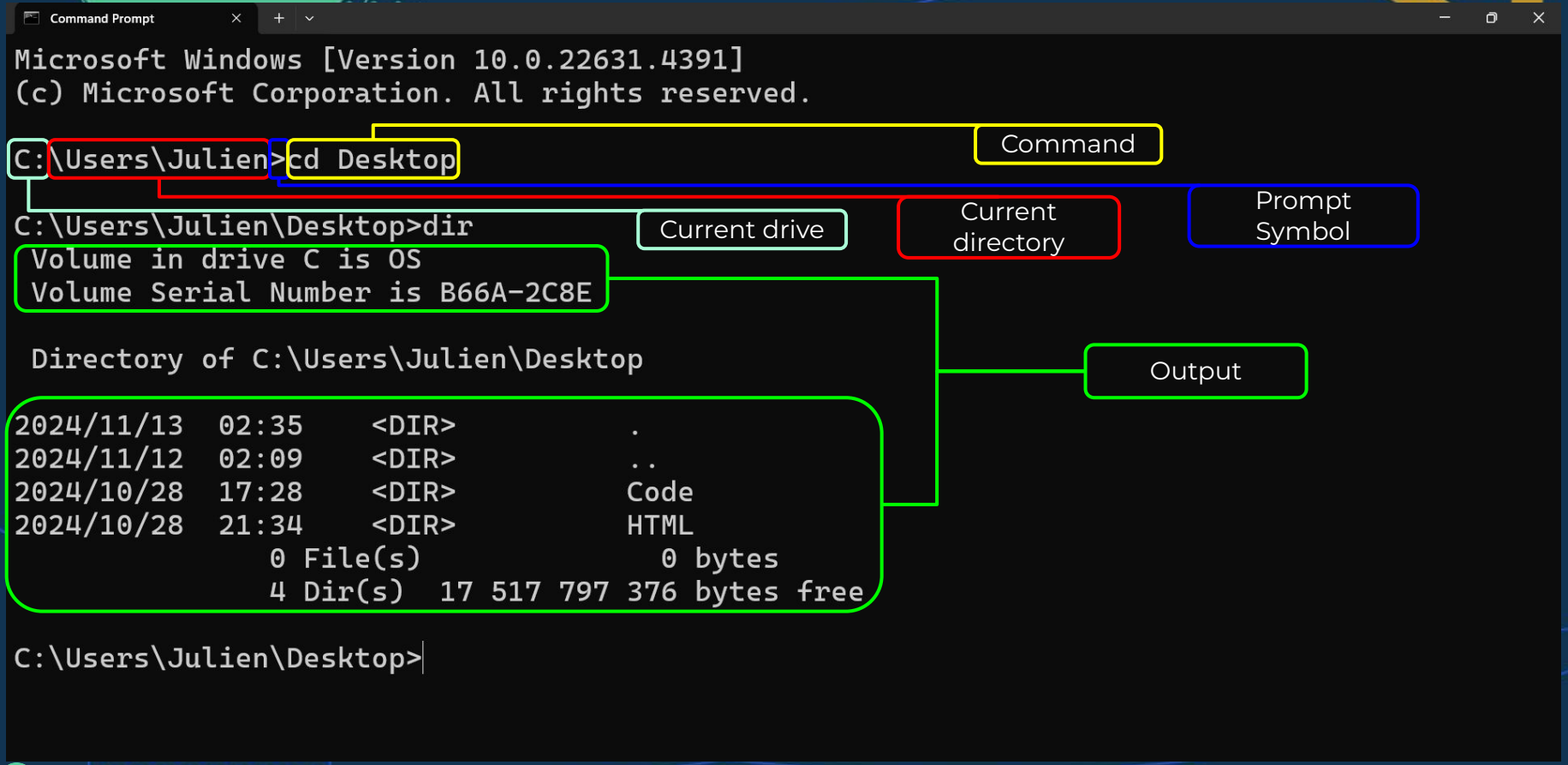
Understanding the Terminal

CoGrammar

What is a Terminal ?

- Definition: A text-based interface that allows users to interact with their computer's operating system by typing commands. Also called CLI or Command Line Interface.
- Why It's Important:
  - Direct Control: Perform actions quickly and efficiently by typing commands instead of using a GUI.
  - Automation: Execute scripts to automate repetitive tasks.
  - Access to System Tools: Use powerful system commands and tools that may not be available in the GUI.

What is a Terminal ?



CoGrammar

Basic Commands

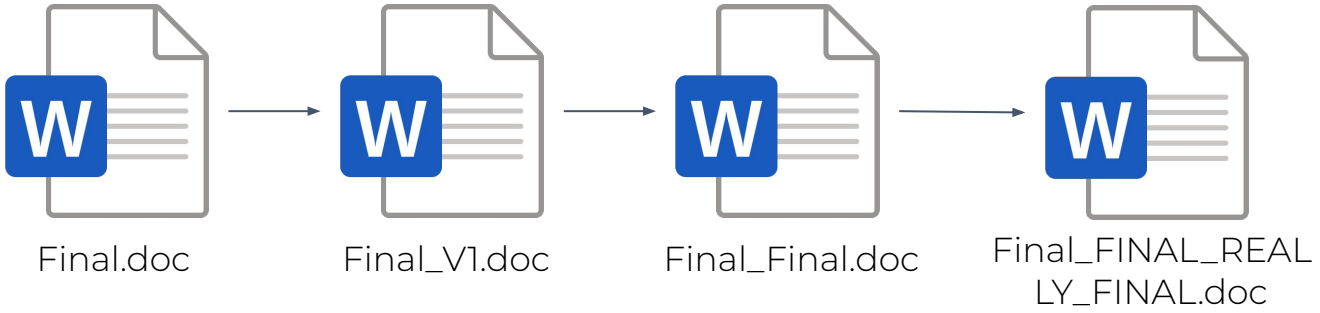
- `pwd` - Prints the current working directory.
- `ls` - Lists the contents of the current directory.
- `cd <directory>` - Changes directory to the specified directory.
- `python filename.py` - Executes a python script.
- `pip install package_name` - Installs Python software packages.
- `mkdir <directory>` - Creates a new directory.
- `touch <file>` - Creates a new file.
- `rm <file>` - Removes a file.

Version Control - Introduction and Fundamentals

CoGrammar

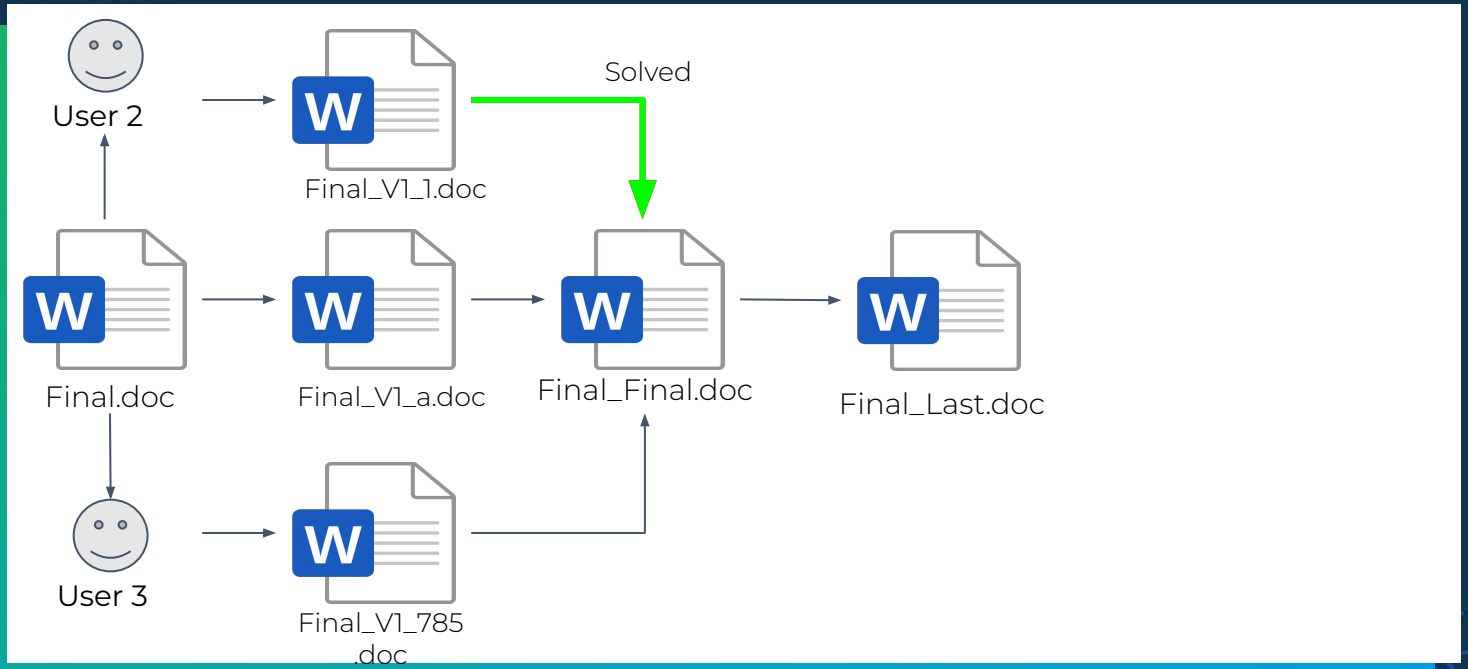
A Common Story... 🤔

Sound familiar?



That's Version Control

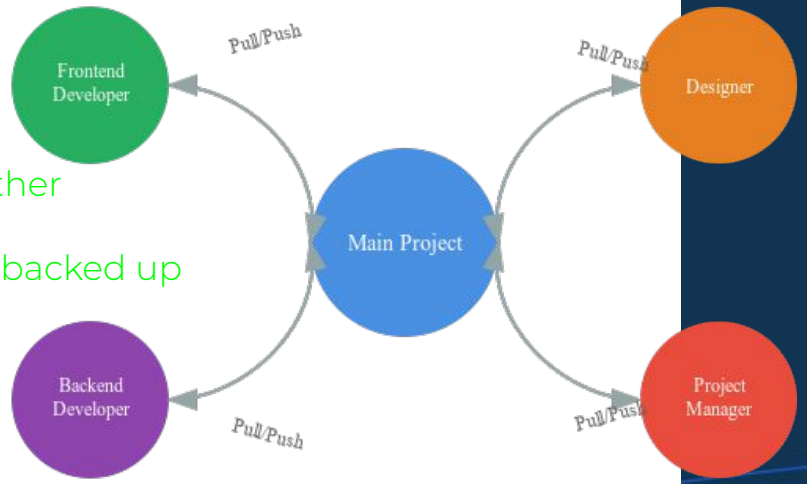
The Chaos of Code 📚



CoGrammar

Problems Solved 💡

- "**I broke the code!**" → Go back in time
- "**Who changed this?**" → Track changes
- "**How do we collaborate?**" → Work together
- "**My computer crashed!**" → Everything's backed up



CoGrammar

The Essentials: Version Control 101

CoGrammar

Meet the Squad: Git Lingo 101!

- **Repository:** The central storage for your project.
- **Commit:** A snapshot of your project at a specific point in time.
- **Branch:** A parallel version of your project.
- **Merge:** Combining changes from different branches.
- **Clone:** Copying an existing repository to your local machine.
- **Working Directory:** Where you make changes to your files.
- **Staging Area:** A temporary holding area for changes before committing.

Theoretical Demo

The Version Control Journey

- **Modified:** Changes made to files in the working directory.
- **Staged:** Files are added to the staging area, preparing them for the next commit.
- **Committed:** Changes are saved in the repository as a new version snapshot.

Branching Out & Coming Back

- **Branching:** Experiment without changing the main code.
- **Merging:** Combine your changes back to the main project.

Theoretical Demo

Let's take a break

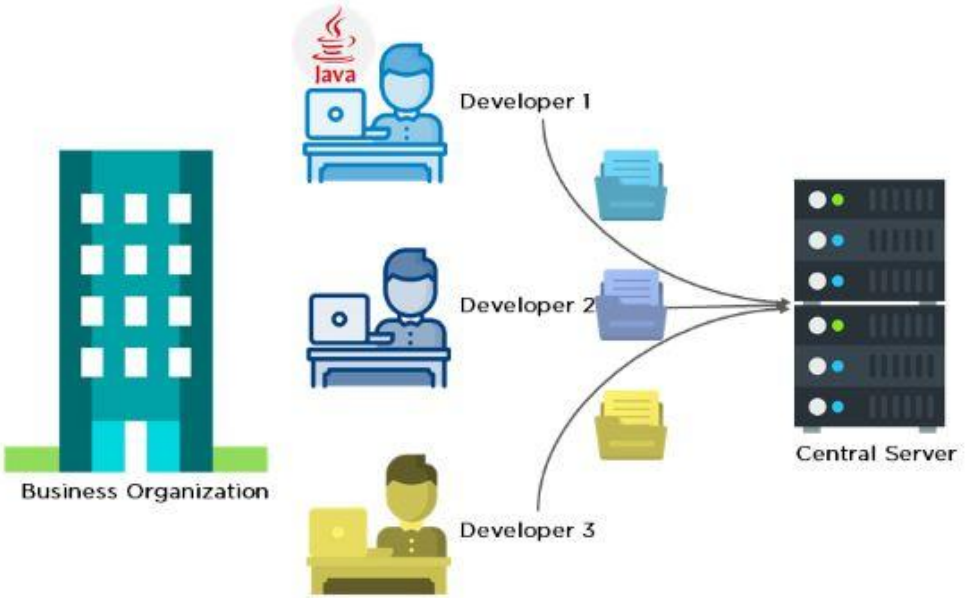
CoGrammar

Introduction to Git

CoGrammar

What is Git? Diclaimer

What is Git?


Java
Developer 1
Developer 2
Developer 3
Business Organization
Central Server

What is Git?


Business Organization
Developer 1
git
Distributed Server
Developer 2
Developer 3

What is Git?

- A powerful version control system
- Tracks changes to your code over time
- Enables collaboration with other developers
- It’s a distributed system, so every developer has a full copy.

Why Git?

- Distributed nature: Work offline and sync later
- Branching and merging: Experiment without risk
- Strong community and support

Your Git Toolkit

- **Repository Operations:**
  - **git init:** Create a new repository
  - **git clone:** Clone an existing repository
- **Working with Changes:**
  - **git status:** Check the status of your files
  - **git diff:** View changes between commits
  - **git add:** Stage changes for commit
  - **git commit:** Commit changes to the repository
  - **git push:** Push changes to a remote repository
  - **git pull:** Pull changes from a remote repository

Real-world Context and Conclusion

CoGrammar

Real-World Git: From Code to Collaboration!

- **Efficient Collaboration:** Teams work together seamlessly on shared codebases.
- **Risk Mitigation:** Backups, version history, and easy rollback.
- **Continuous Integration/Continuous Delivery (CI/CD):** Automated testing and deployment.
- **Open Source Development:** Fostering community and collaboration.

The Big Three: Where the Magic Happens!

GitHub

GitLab

Bitbucket

Best Practices: Keep Calm and Commit On!

- **Commit Frequently:** Small, focused commits.
- **Write Clear Commit Messages:** Describe the changes made.
- **Use Branches Effectively:** Isolate features and bug fixes.
- **Review Code Regularly:** Improve code quality and collaboration.
- **Utilize Pull Requests:** A structured review process.
- **Automate Your Workflow:** Use CI/CD pipelines.

Poll

What is a repository in version control?

1. A folder that contains only images
2. A backup drive for your computer
3. A tool for writing code
4. A storage location for your project and its version history

Poll

What is the correct sequence of steps for saving changes in Git?

1. Commit → Add → Push
2. Push → Add → Commit
3. Commit → Push → Add
4. Add → Commit → Push

Lesson Conclusion and Recap

Recap the key concepts and techniques covered during the lesson.

- **Operating the Terminal**: Demonstrate basic command-line navigation using pwd, cd, and ls to explore files and directories. Explain the command prompt and how to modify directory structures through terminal commands.
- **Defining Version Control**: Explain version control purpose and analyse its necessity in software development.
- **Applying Git Terminology**: Define essential Git concepts and illustrate terms like repository, commit, branch, merge, working directory, and staging area. Classify each team's role in version control workflow.
- **Executing Basic Git Commands**: Identify and describe fundamental Git commands for version control. List and differentiate between commands like git init, clone, status, diff, add, commit, push, and pull, demonstrating their purpose and basic usage.
- **Relating Professional Context**: Describe how version control integrates into professional software development. Recognize platforms like GitHub, examine industry practices, and justify the importance of version control in team collaboration.

Practical: Basic Git and GitHub Setup and Workflow

1. **Objective:** Create a repository, make changes, and understand how to push these changes to GitHub. This exercise will help to have a basic understanding of how to initialise a repository, make commits, and work with GitHub as a remote.
2. **Steps to Implement:**
    - Set Up and Initialize Git
    - Create a Local Repository
    - Create and Add Files to the Repository
    - Commit Changes
    - Push Changes to GitHub
    - Verify and View Changes on GitHub

Resources

- **Additional Resources**
  - 1.5 Getting Started - Installing Git
  - Pro Git book

Questions and Answers

CoGrammar

Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar