Welcome to this CoGrammar Lecture: Git Workflow and Collaboration

The session will start shortly...

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

CoGrammar

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

CoGrammar

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

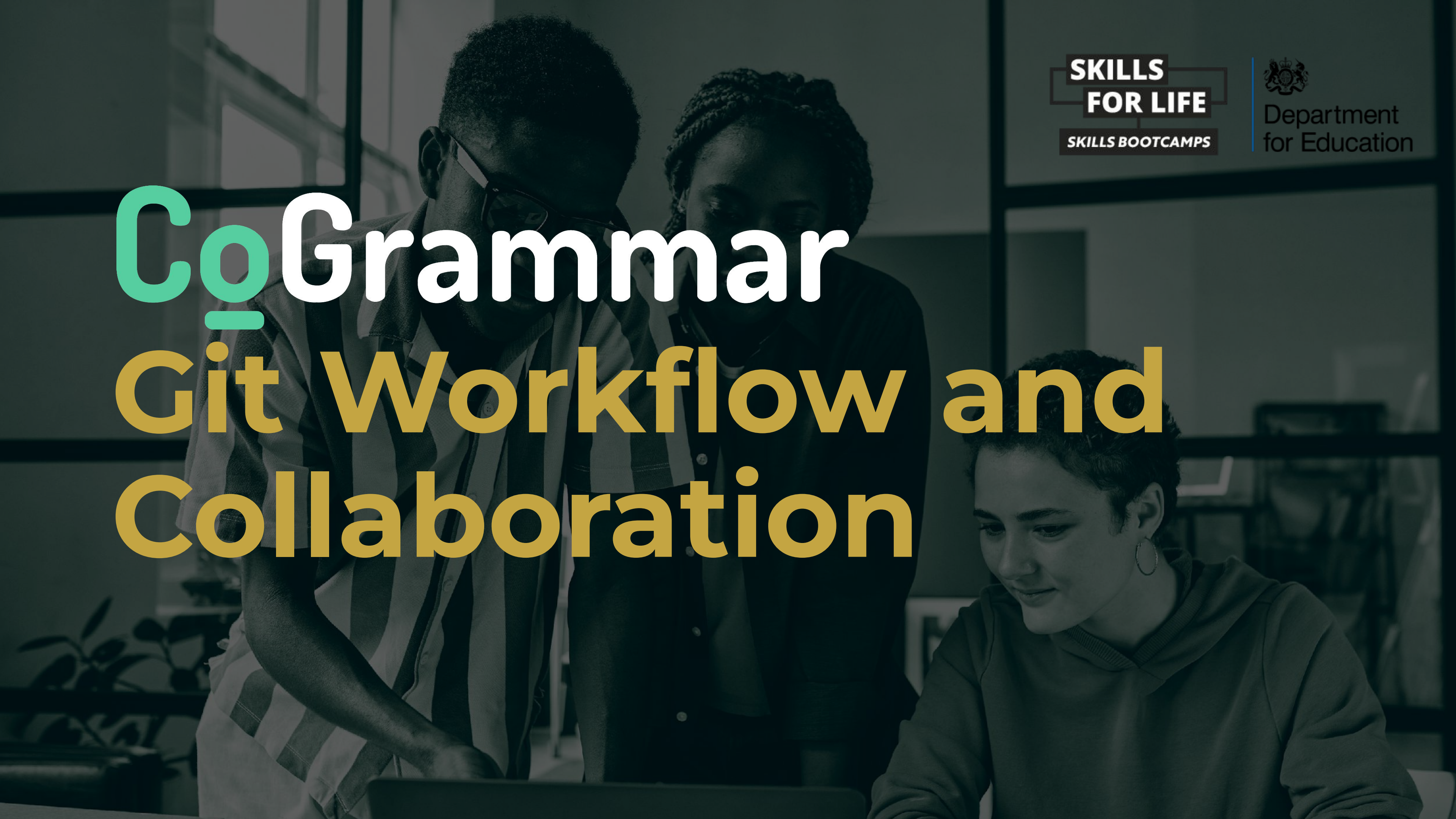
Ronald Munodawafa

Tevin Pitts

Scan to report a safeguarding concern

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

CoGrammar HyperionDev



SKILLS FOR LIFE
SKILLS BOOTCAMPS
Department for Education

CoGrammar

Git Workflow and Collaboration

Learning Objectives & Outcomes

- Perform Git Collaboration and Best Practices
- Explain branching strategies (feature branches, master/main)
- Describe merging branches and resolving conflicts
- Explain and describe .gitignore
- Explain and perform to Pull Requests (PRs) and code reviews (via GitHub)
- Resolve merge conflicts effectively.
- Assess the impact of version control on collaboration.
- Collaborate on a shared project using remote repositories and platforms like GitHub.

CoGrammar

Git Workflow & Collaboration in Enterprise

CoGrammar

Git Workflow & Collaboration in Enterprise

Google Android Development

- 2000+ developers
- 40+ million lines of code
- 40,000+ commits monthly
- Git-based version control essential

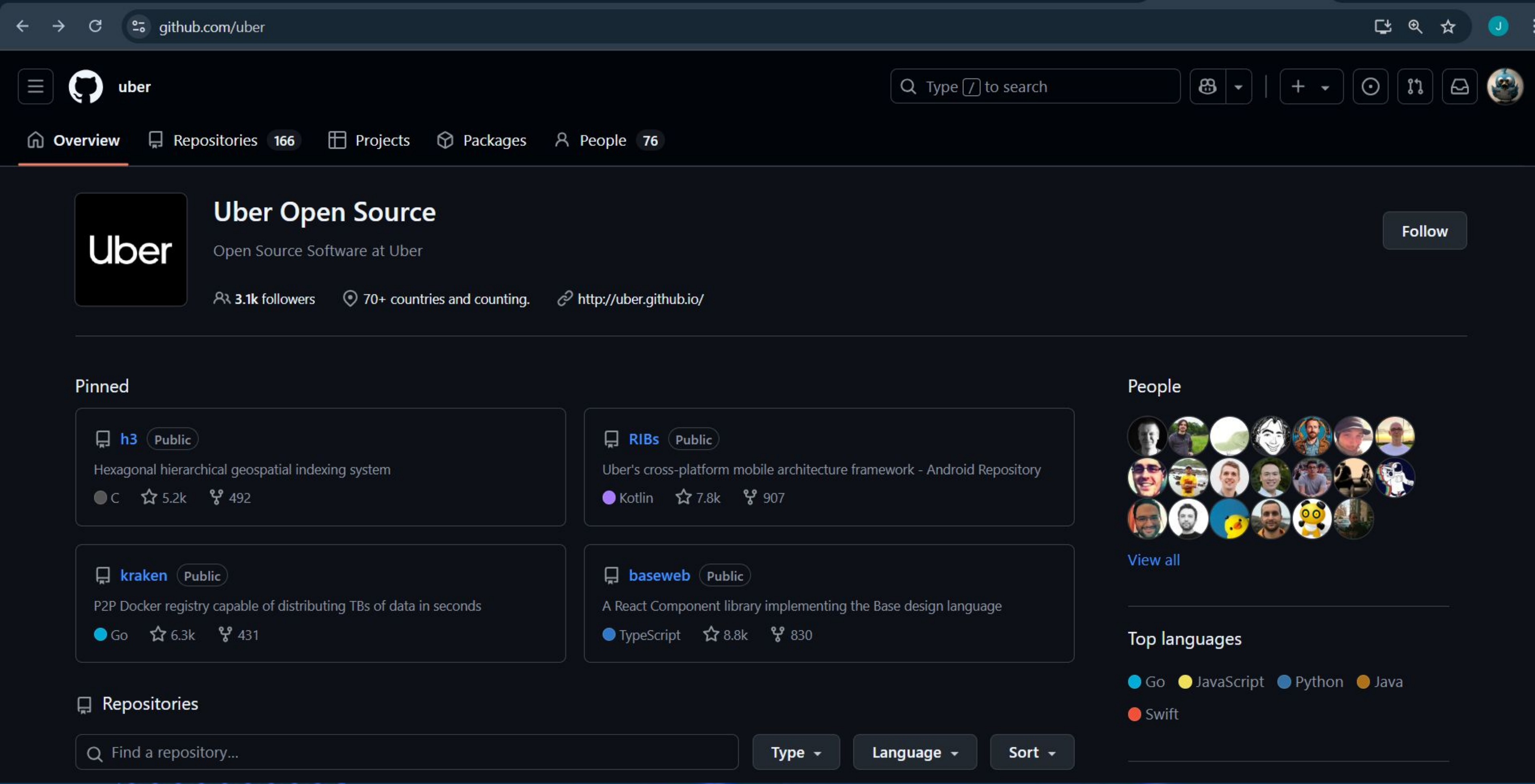
Microsoft's Windows Development

- World's largest Git repo
- 4000+ developers
- 85+ million lines of code
- Git enables simultaneous development

Why This Matters:

- Modern software development requires coordinating thousands of developers
- Version control is crucial for managing complex codebases and preventing conflicts
- Git skills are essential for working in any modern development team
- Understanding Git workflow is a fundamental skill for career growth in tech

CoGrammar



github.com/uber

uber

Type / to search

Overview Repositories 166 Projects Packages People 76

Uber Open Source

Open Source Software at Uber

3.1k followers 70+ countries and counting. http://uber.github.io/

Follow

Pinned

h3 Public
Hexagonal hierarchical geospatial indexing system
C 5.2k 492

RIBs Public
Uber's cross-platform mobile architecture framework - Android Repository
Kotlin 7.8k 907

kraken Public
P2P Docker registry capable of distributing TBs of data in seconds
Go 6.3k 431

baseweb Public
A React Component library implementing the Base design language
TypeScript 8.8k 830

Repositories

Find a repository... Type Language Sort

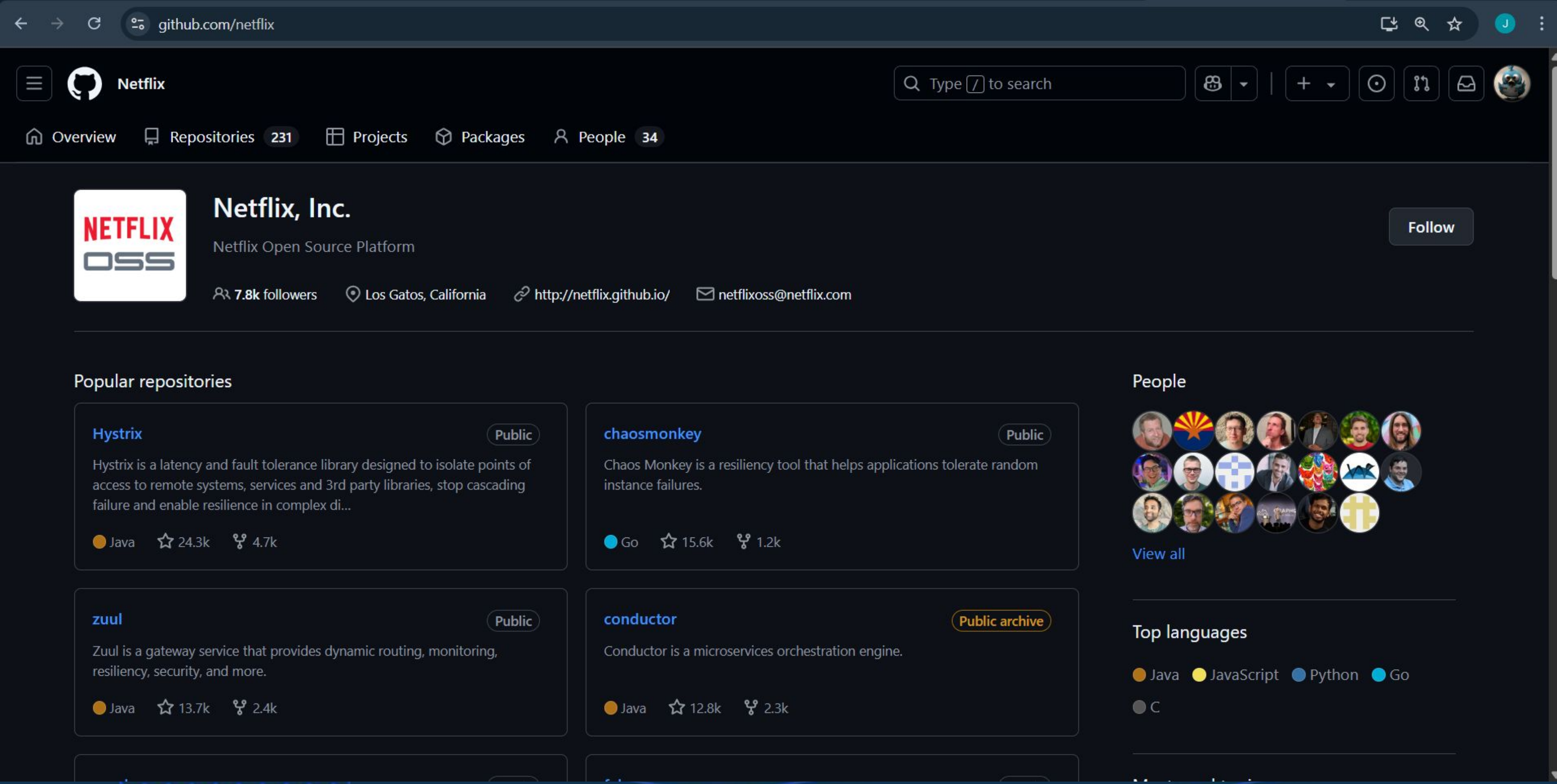
People

View all

Top languages

Go JavaScript Python Java Swift

CoGrammar



github.com/netflix

Netflix

Type / to search

Overview Repositories 231 Projects Packages People 34

Netflix, Inc.

Netflix Open Source Platform

7.8k followers Los Gatos, California http://netflix.github.io/ netflixoss@netflix.com

Follow

Popular repositories

Hystrix Public
Hystrix is a latency and fault tolerance library designed to isolate points of access to remote systems, services and 3rd party libraries, stop cascading failure and enable resilience in complex di...
● Java ☆ 24.3k 4.7k

chaosmonkey Public
Chaos Monkey is a resiliency tool that helps applications tolerate random instance failures.
● Go ☆ 15.6k 1.2k

zuul Public
Zuul is a gateway service that provides dynamic routing, monitoring, resiliency, security, and more.
● Java ☆ 13.7k 2.4k

conductor Public archive
Conductor is a microservices orchestration engine.
● Java ☆ 12.8k 2.3k

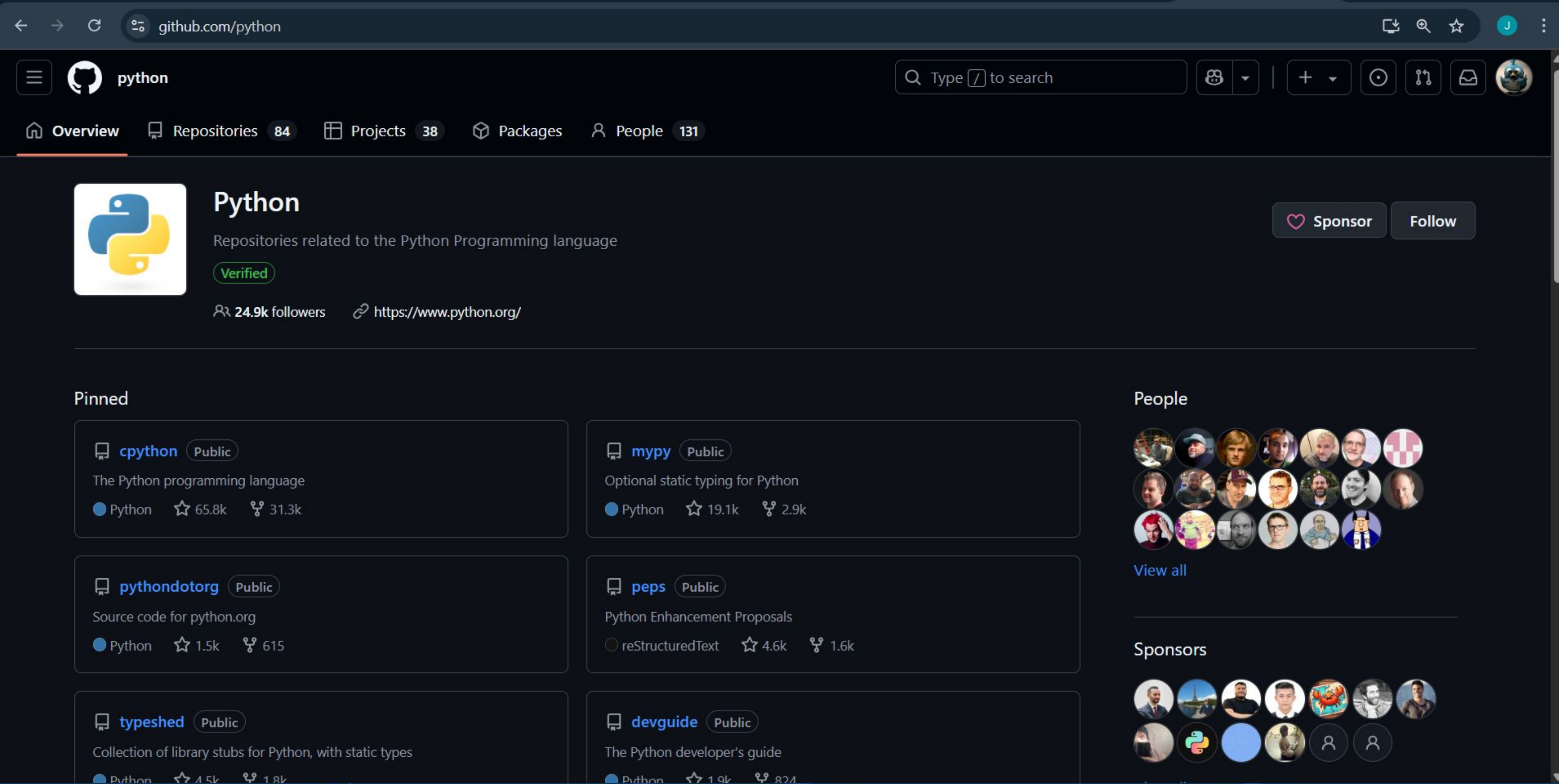
People

View all

Top languages

● Java ● JavaScript ● Python ● Go ● C

CoGrammar



github.com/python

python

Type / to search

Overview · Repositories 84 · Projects 38 · Packages · People 131

Python

Repositories related to the Python Programming language

Verified

24.9k followers · https://www.python.org/

Sponsor · Follow

Pinned

cpython Public
The Python programming language
Python 65.8k 31.3k

mypy Public
Optional static typing for Python
Python 19.1k 2.9k

pythondotorg Public
Source code for python.org
Python 1.5k 615

peps Public
Python Enhancement Proposals
reStructuredText 4.6k 1.6k

typeshed Public
Collection of library stubs for Python, with static types
Python 4.5k 1.8k

devguide Public
The Python developer's guide
Python 1.9k 824

People

View all

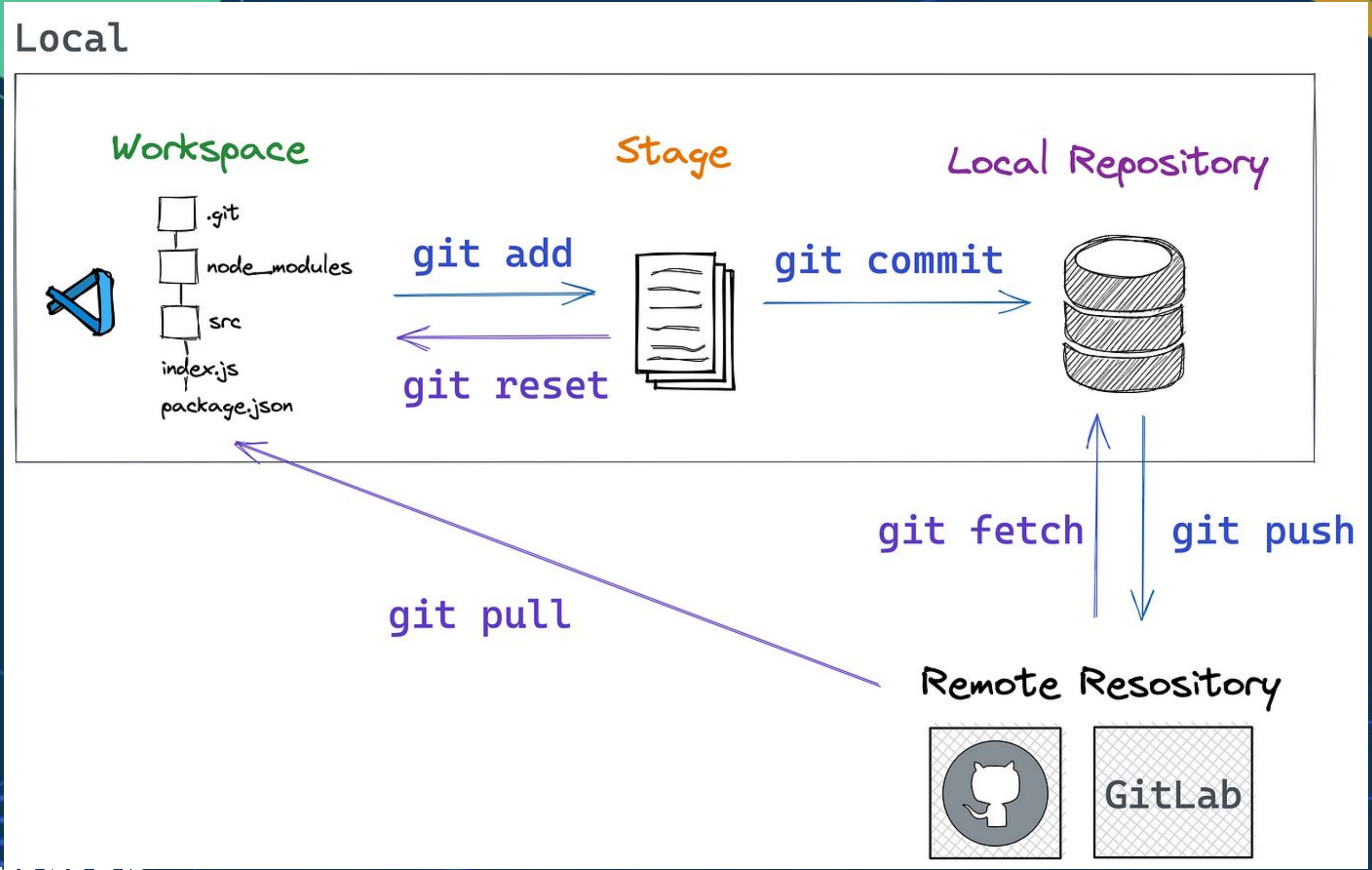
Sponsors

CoGrammar

Introduction

CoGrammar

Recap



CoGrammar

Branching Strategies

CoGrammar

What Are Branches in Git?

- **What is a branch?**
  - A branch is like a separate workspace in your project where you can work without affecting the main code.
- **Purpose of the Main/Master Branch:**
  - The *main branch* (or master) is the stable, production-ready version of your project.
  - Changes merged here should always work as intended.
- **Why Protect the Main Branch?**
  - Prevent accidental changes by requiring reviews before merging.
  - Keep the project stable for releases or deployments.

CoGrammar

Feature Branches: Work Without Worry

- **What is a Feature Branch?**
    - A branch dedicated to developing a specific feature or fix.
    - Allows independent work without affecting others.
- **Why Use Feature Branches?**
    - Keep the main branch stable.
    - Organize your work better.
- **Branch Lifecycle:**
    - Create → Work → Test → Merge → Delete

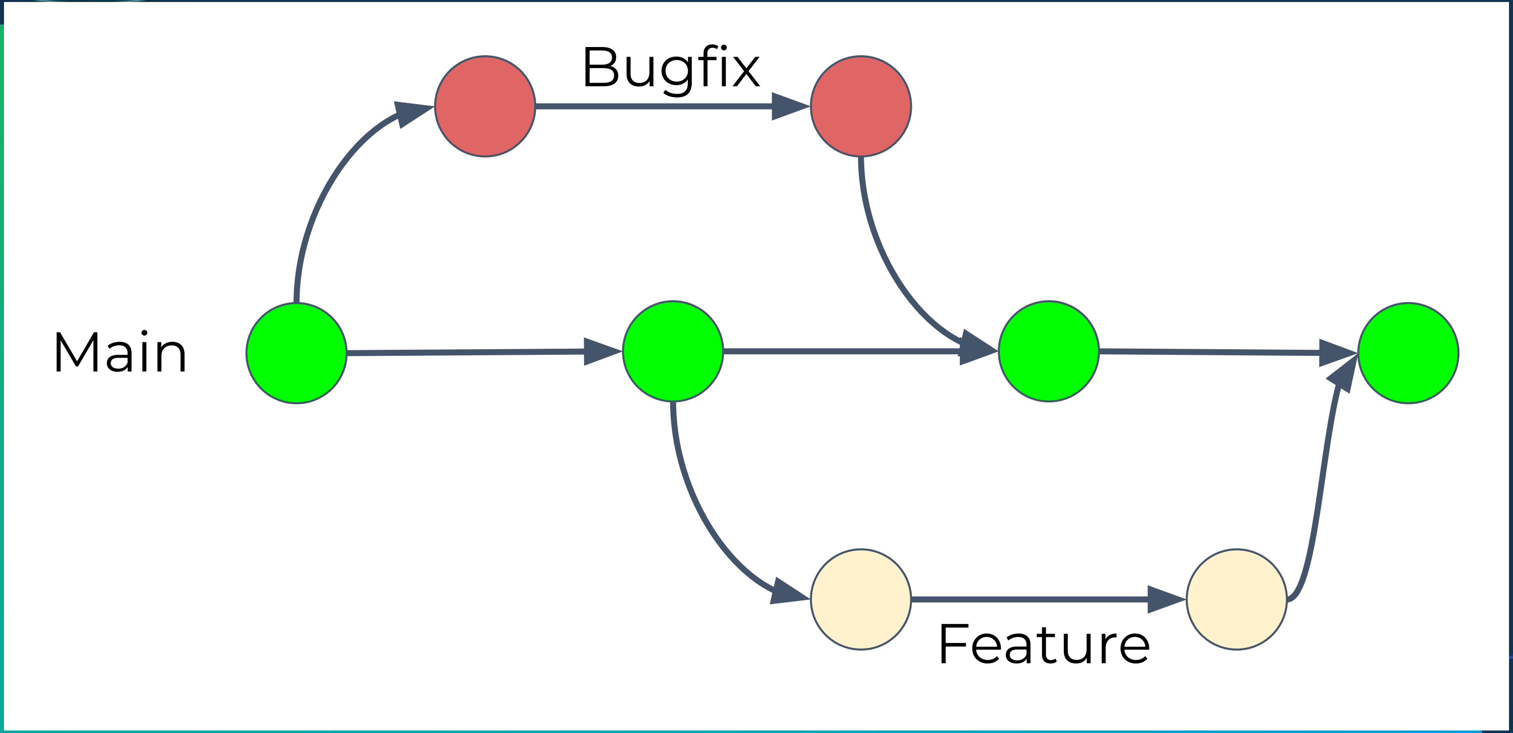
CoGrammar

Feature Branches: Work Without Worry

- **Good Naming Practices:**
  - Be descriptive but concise.
  - Use formats like:
    - `feature/new-login-page`
    - `bugfix/fix-login-issue`
    - `hotfix/urgent-deploy-fix`
- **Why Naming Matters:**
  - Clear names help the team understand the branch’s purpose.
  - Avoids confusion in collaborative projects.
- **Tip:**
  - Use lowercase and dashes (`-`) for readability.
- Example list of well-named branches vs. poorly named branches:
  - ✅ `feature/add-user-auth`
  - ❌ `1234` or `newbranch`.

CoGrammar

Branching Strategies: A Visual Guide



CoGrammar

Collaboration Fundamentals

CoGrammar

Understanding Remote and Local Repositories

- **What is a Remote Repository?**
  - A version of your repository hosted online (e.g., GitHub).
  - Accessible by your entire team, enabling collaboration.
- **Local vs Remote:**
  - *Local*: Stored on your computer.
  - *Remote*: Shared on a platform for teamwork.
- **Why Use GitHub? (Or GitLab, Bitbucket).**
  - Easy sharing, and collaboration tools, collaborate with others, back up your code, track changes over time.

CoGrammar

What is a Pull Request?

- **Purpose of PRs:**
  - A formal request to review and merge your work into a shared branch.
  - Encourages collaboration and ensures quality control.
- **When to Create a PR?**
  - After completing a feature or fix in your branch.
- **PR Workflow:**
  1. Submit a PR on GitHub.
  2. Request reviews from team members.
  3. Address feedback and make changes if needed.
  4. Merge your branch when approved.

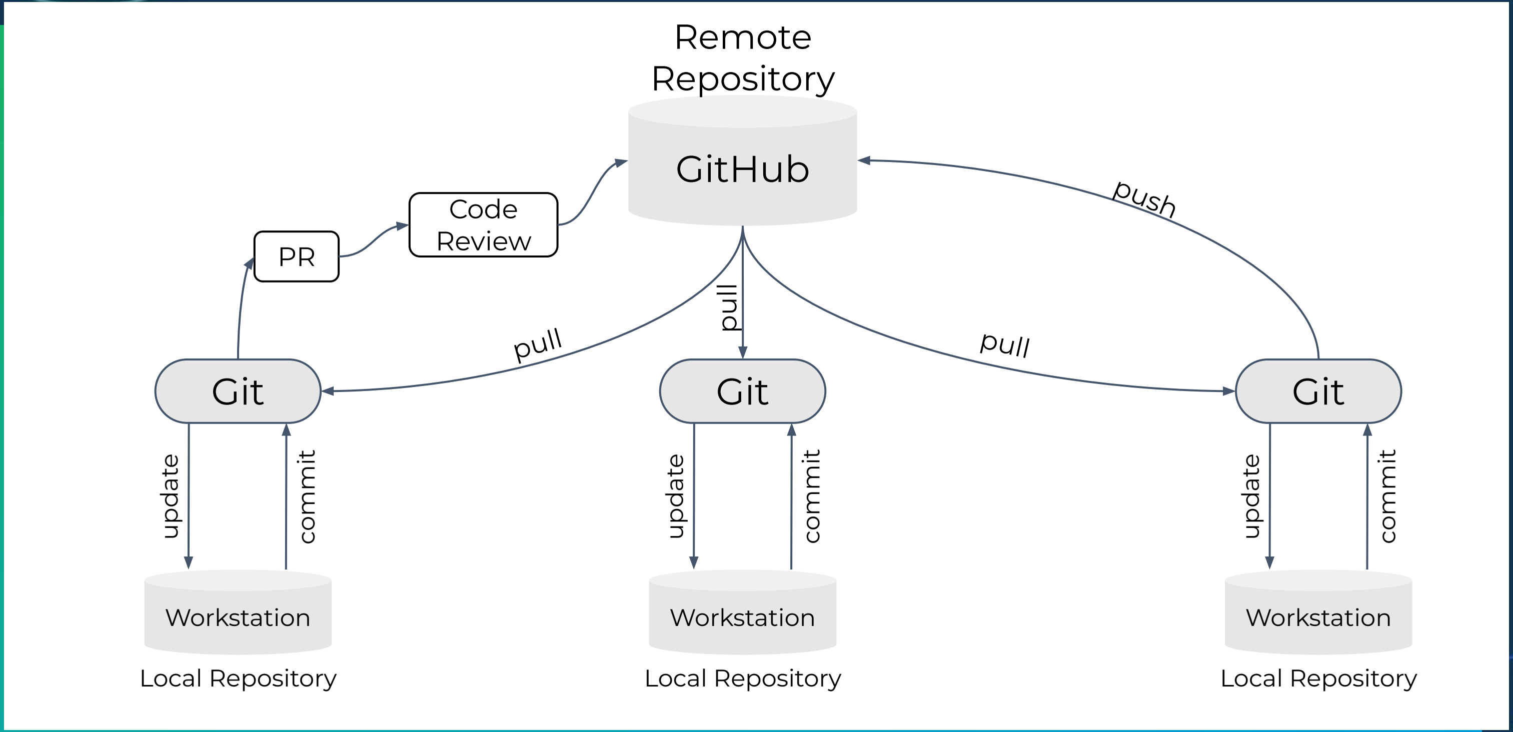
CoGrammar

Code Reviews: Building Better Software Together

- **Why Code Reviews Matter:**
  - Catch bugs or issues early.
  - Ensure adherence to team coding standards.
  - Encourage knowledge sharing among team members.
- **Writing Effective PR Descriptions:**
  - Be clear and concise.
  - Describe ***what*** you did and ***why*** you did it.
  - Mention any specific areas needing attention during the review.
- **Best Practices for PRs:**
  - Small, focused changes (don’t lump multiple features).
  - Respond to feedback promptly.

CoGrammar

Pull Requests and Code Reviews



CoGrammar

Merge Operations & Conflict Resolution

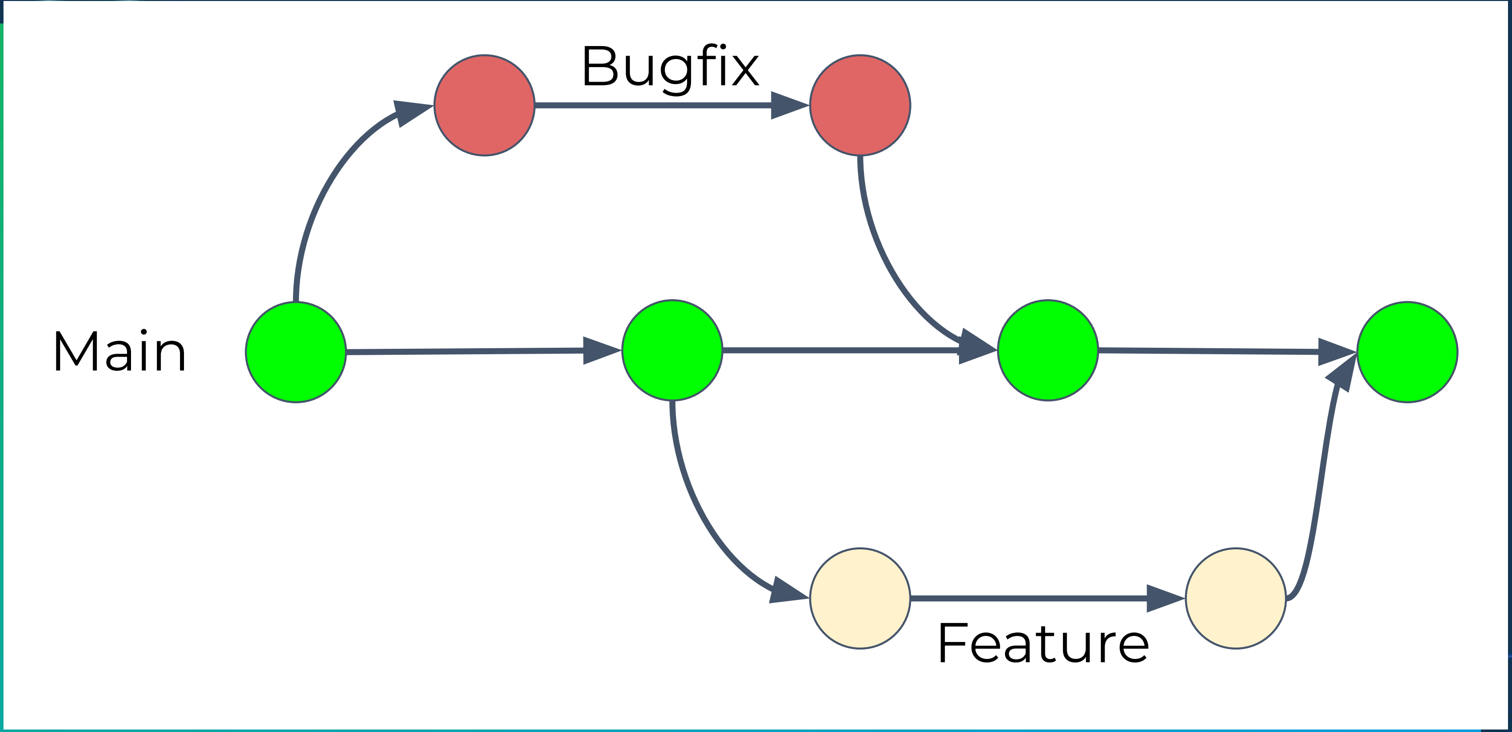
CoGrammar

What is a Merge and Why Do We Need It?

- **What is a Merge?**
    - Combining changes from one branch into another.
    - Ensures all features and fixes come together.
- **When to Merge:**
    - After completing a feature or fix in a feature branch.
    - Before a release to ensure all work is integrated.

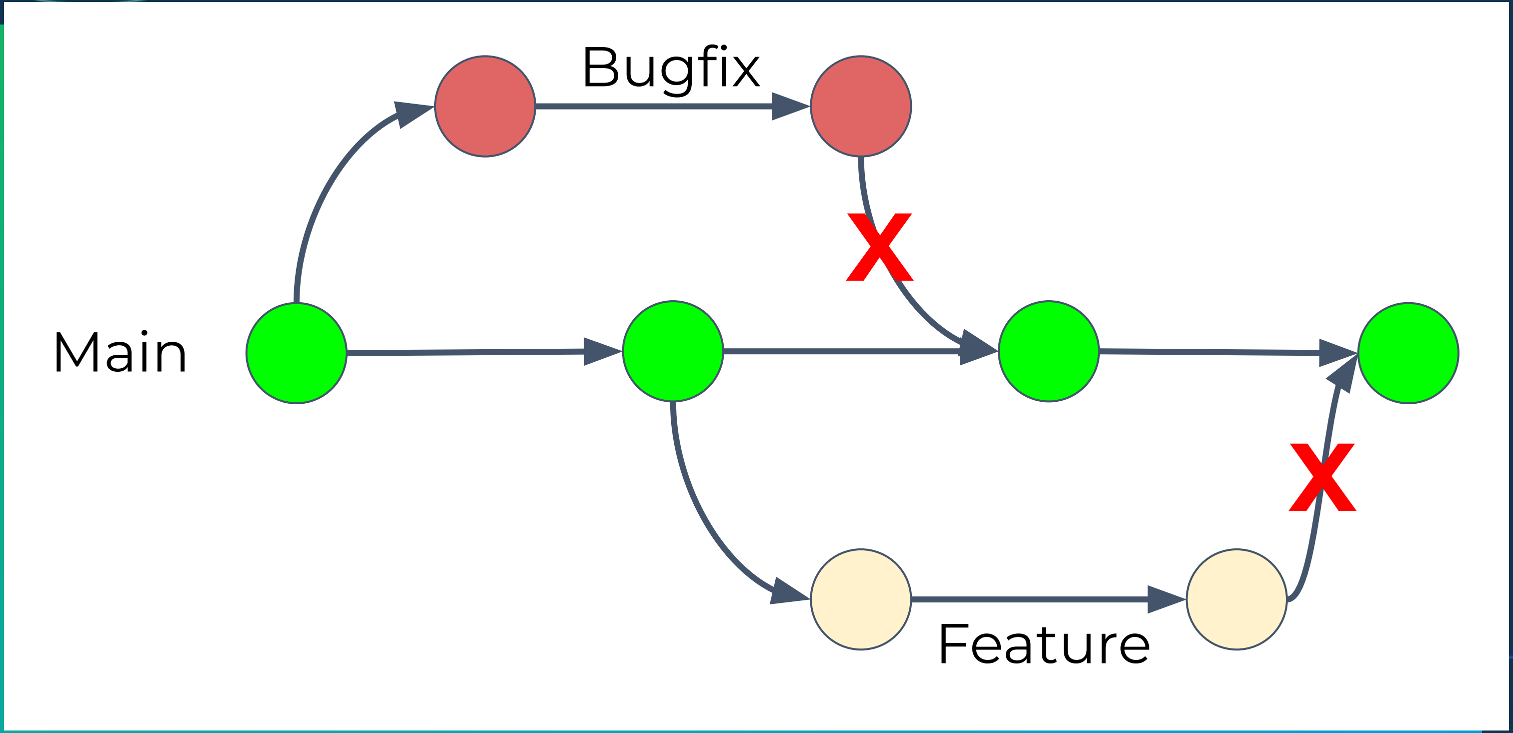
CoGrammar

What Are Branches in Git?



CoGrammar

Handling Merge Conflicts



CoGrammar

Handling Merge Conflicts

- **Why Do Merge Conflicts Happen?**
    - Two branches modify the same line in a file.
    - Changes are made to the same file in ways Git cannot automatically combine.
- **How to Resolve Conflicts:**
    1. Identify the conflict (Git will mark files with conflicts).
    2. Edit the file to keep the desired changes.
    3. Mark the conflict as resolved (`git add`).
    4. Complete the merge (`git commit`).
- **Tools to Simplify Conflict Resolution:**
    - VS Code Git integration.
    - GitHub's conflict resolution editor.

CoGrammar

Best Practices & Tools

CoGrammar

Managing What Git Tracks with .gitignore

- **What is .gitignore?**
  - A file to tell Git which files or directories to ignore (not track).
  - Keeps sensitive or irrelevant files out of the repository.
- **Common Use Cases:**
  - Ignoring files like logs, temporary files, and environment variables.
  - Excluding OS or editor-specific files (e.g., `.DS_Store`, `*.swp`).
- **How to Use .gitignore:**
  - Add a `.gitignore` file at the root of your project.
  - Use patterns to specify ignored files (e.g., `*.log`, `/node_modules/`).
- **Best Practice:**
  - Always add `.gitignore` when initializing a project to avoid tracking unnecessary files.

CoGrammar

Guidelines for Effective Code Reviews

- **What is Code Review?**
  - A systematic examination of code by peers to improve quality and ensure adherence to team standards.
- **Review Guidelines:**
  - Focus on the code, not the person.
  - Check for functionality, readability, and adherence to standards.
  - Ensure the code is well-tested.
- **Providing Constructive Feedback:**
  - Be specific: "Consider renaming this variable to make it clearer."
  - Be polite: "What if we refactor this function for better readability?"
  - Avoid negative or personal comments.

CoGrammar

Using GitHub's Review Features Effectively

- **GitHub Review Features:**
  - Leave inline comments on specific lines of code.
  - Approve or request changes on pull requests.
  - Use suggestions for quick fixes.
- **Best Practices for Reviewers:**
  - Understand the feature's purpose before reviewing.
  - Test locally if necessary.
  - Avoid nitpicking minor issues unless they impact functionality.
- **Best Practices for Submitters:**
  - Write clear commit messages.
  - Use meaningful PR descriptions (what/why/how).
  - Address feedback promptly and update your PR.

CoGrammar

Git GUI Tools

GIT EXTENSIONS

GitKraken

https://git-scm.com/downloads/guis

Lesson Conclusion and Recap

Recap the key concepts and techniques covered during the lesson.

- **Branching Strategies:** Key strategies like feature branches and main branch usage help organise work and streamline collaboration.
- **Merging and Conflict Resolution:** Merging branches and handling merge conflicts ensure smooth integration of changes from different contributors.
- **Using .gitignore:** The .gitignore file helps manage which files to track or ignore, keeping the repository clean and focused.
- **Pull Requests (PRs) and Code Reviews:** PRs and code reviews support collaborative development, allowing team members to review, discuss, and improve code before merging.
- **Remote Repositories and GitHub Collaboration:** Leveraging platforms like GitHub enhances teamwork, making it easier to share, collaborate, and track project progress.

CoGrammar

Resources

Resources

- **Software:**
  - Git - Downloading Package
  - Download GitHub Desktop
- **Additional Resources**
  - Hello World - GitHub Docs
  - Get started with GitHub documentation
- **Books:**
  - Pro Git book

CoGrammar

Questions and Answers

CoGrammar

Thank you for attending

SKILLS FOR LIFE

SKILLS BOOTCAMPS

Department for Education

CoGrammar