



**Welcome to
lecture 11!**

Agenda

Session Objectives

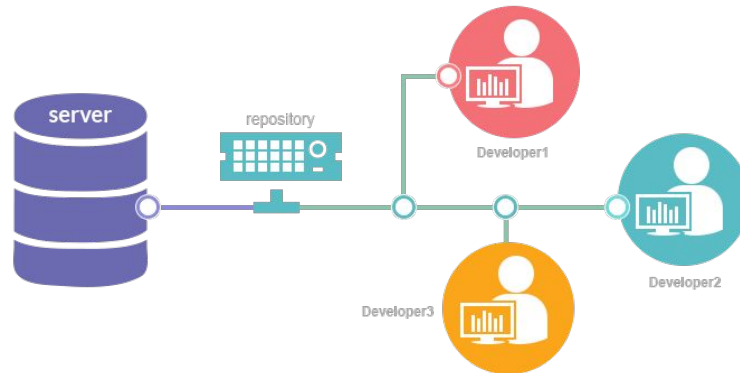
- Challenges in managing software
 - What's the solution?
- A brief history of Git
 - Basics of Git commands
 - Demo!
- What's Github?
 - How's Github different from Git?
 - Demo!
- Wrap up the website
- Quiz
- Homework

The Problem: Change is hard!

- Managing changes to files manually can be messy
- Have you ever ended up with multiple versions of the same file?
 - report_final.doc
 - report_final_v2.doc
 - report_really_final_this_time_v3.doc
- Challenges with this approach
 - Which version is the correct one?
 - What changed between the version?
 - If I made a mistake in report_final_v2, how can I undo it?
 - How to collaborate without overwriting changes?

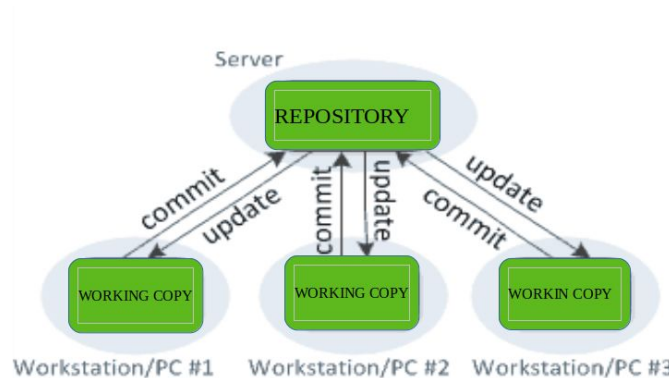
The Solution: Version Control System

- What is a version control system (vcs)?
 - Software that helps you manage changes to files or projects over time
 - Think of it like a "time machine" for your work
- Key Benefits
 - Tracks history: Who changed what, and when?
 - Revert: Go back to previous versions easily
 - Understand changes: Compare different versions
 - Collaboration: Enables teams to work together effectively
- Types of VCS
 - Centralized VCS
 - Distributed VCS



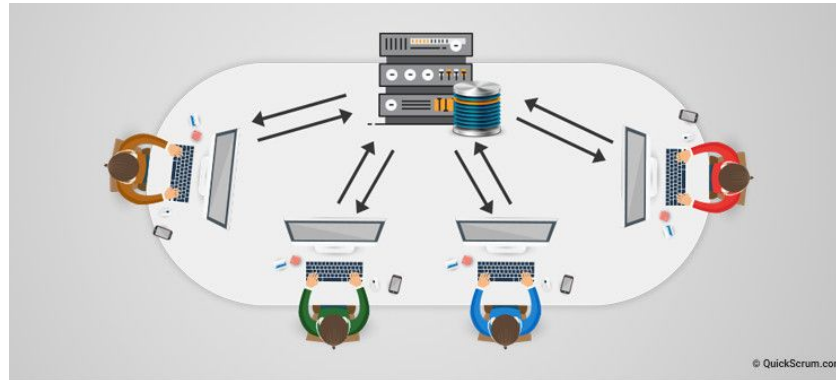
A Glimpse at the Past: Centralized VCS

- What's a centralized VCS?
 - One main central server holds all the project history
 - Team members "check out" files from the server and "check in" changes
- Software like Subversion (SVN) & Concurrent Versions System (CVS)
- Limitations
 - Single point of failure (if the server is down, work stops)
 - Often requires network connection for most operations
 - Branching and merging could be slow and complex



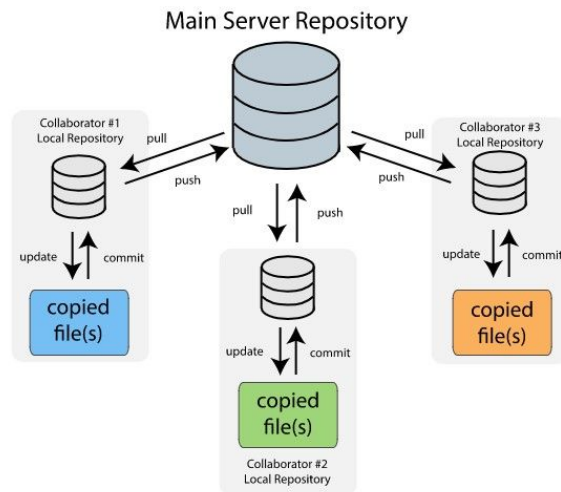
Birth of Git: A Quick Story

- Created by Linus Torvalds (also creator of Linux) in 2005
- Developed to manage the massive, distributed development of the Linux kernel
- Key Requirements when building Git
 - Speed: Faster operations were crucial
 - Distributed: Remove dependency on a single central server. Allow offline work
 - Better Branching/Merging: Support non-linear workflows and parallel development easily
 - Data Integrity: Ensure history couldn't be easily corrupted
- Let's understand the core concepts



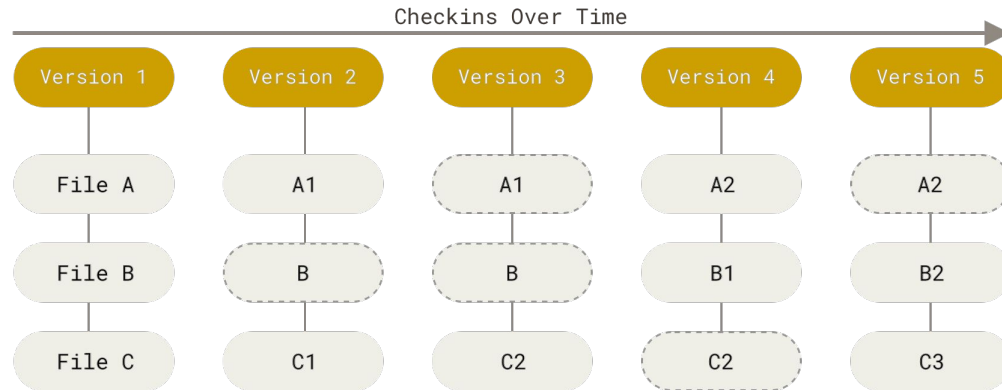
Core Concepts: Distributed

- Unlike centralized systems, Git is Distributed
- When you **clone** a repository, you get a full copy of the *entire history* on your local machine.
- Benefits
 - Work offline (commit, view history, branch without internet)
 - Faster operations (most actions are local)
 - No single point of failure (everyone has a full backup)



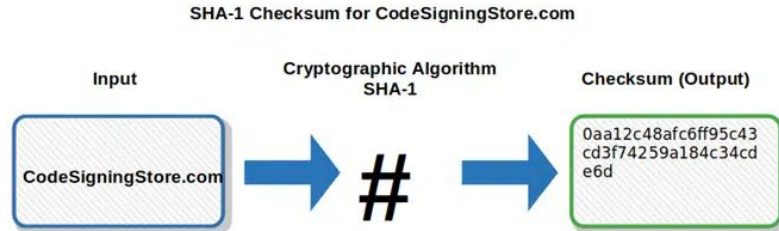
Core Concepts: Snapshots, Not Differences

- Most other systems store information as a list of file-based changes
 - Subversion, CVS & others think of information they store as a set of files and changes made to each file over time. Also known as delta-based version control
- Git thinks about data like a series of **snapshots** of your entire project
- When you commit
 - Git takes a picture of what all your files look like at that moment
 - If files haven't changed, Git just links to the identical stored file. This is a major efficiency gain!



Core Concepts: Integrity

- Everything in Git is **checksummed** before it's stored
 - A checksum is like a unique seal for data - that seal is called a secure hash
- Git uses a cryptographic hash function called SHA-1 to calculate these checksums
 - Cryptographic hash function: like a digital fingerprint—it takes any input data and creates a unique, fixed-size output that cannot be reversed, ensuring security and integrity
 - A SHA-1 hash looks like this: e2c4c9026edd38a6919a2e3eb0a758586366b0fc
- Result
 - Impossible to change file contents or history without Git knowing
 - Protects against accidental corruption or data loss
 - Ensures the history you see is the real history

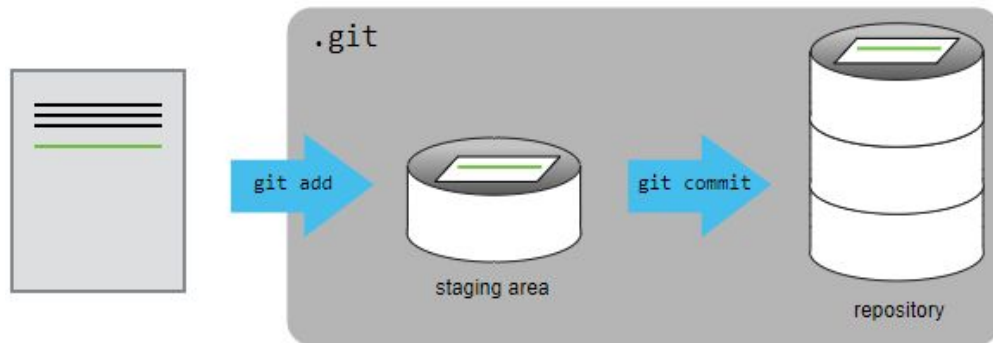


Your Local Workflow: The Three Areas

- Git manages files in three main areas on your local machine
 - Working Directory: The actual files you see and edit in your project folder
 - Staging Area (Index): An intermediate area where you prepare ("stage") changes for the next commit
 - Repository (.git directory): Where Git permanently stores the history (committed snapshots)
- When you commit
 - Working Directory: The actual files you see and edit in your project folder
 - If files haven't changed, Git just links to the identical stored file. This is a major efficiency gain!
- This is different from systems that primarily store differences between files

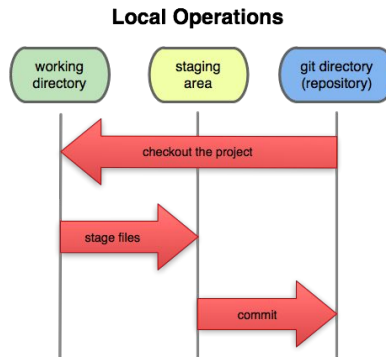
Understanding the Staging Area (Index)

- What is the staging area?
 - Intermediate space where changes are collected before they are committed
 - Think of it like a shopping cart, a loading dock, or a draft email
- Why does it exist?
 - Allows you to carefully craft your commits
 - You can choose which changes to include
- How does git know what's in the staging area?
 - A file (`.git/index`) that lists what will go into your next commit



Basic Git Cycle: Modify -> Stage -> Commit

- What is the staging area?
 - Unstaged: Make changes to files in your Working Directory
 - git status will help identify the unstaged files
 - Stage: Choose which changes you want in the next commit and add them to the Staging Area
 - git add is used to add the file to the staging area
 - Commit: Save the snapshot from the Staging Area permanently into the Repository
 - git commit is used to commit the changes



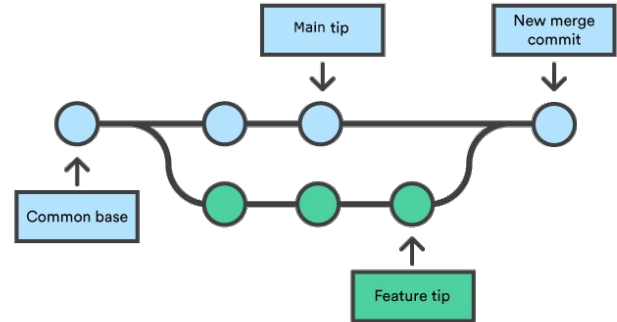
Branching & Merging: Parallel Development

- Branching

- Allows you to create separate lines of development
- Essential for
 - Working on new features without affecting the main codebase
 - Experimenting with different approaches
 - Fixing bugs in isolation
- Commands: git branch, git checkout

- Merging

- Combines changes from one branch into another
- Essential for integrating new features or bug fixes
- Command: git merge. Advanced: git rebase





Demo Time!







Shifting Gears: What's Github?



What is GitHub?

- Popular web-based platform for hosting Git repositories
 - A service build *around* git
 - There are others in the space: GitLab, GitBucket & more
- Think of it as
 - Cloud storage designed for Git projects
 - A place to backup your local work
 - A platform for collaboration and sharing code
- A project on Github is usually called a “repository”


Refactor all the things #161



 Merged mergery merged 1 commit into `main` from `refactor`  now



 Conversation 1  Commits 1  Checks 0  Files changed 1

 adamu commented 14 minutes ago Owner 


No description provided.


 Refactor all the things

 adamutest approved these changes  continuously integrating... new changes

 adamutest left a comment Collaborator 

LGM if the tests pass! 🍌

 adamu added the `automerge` label 1 minute ago

 mergery bot merged commit `4becd88` into `main` now
1 check passed View details Revert

Git vs Github

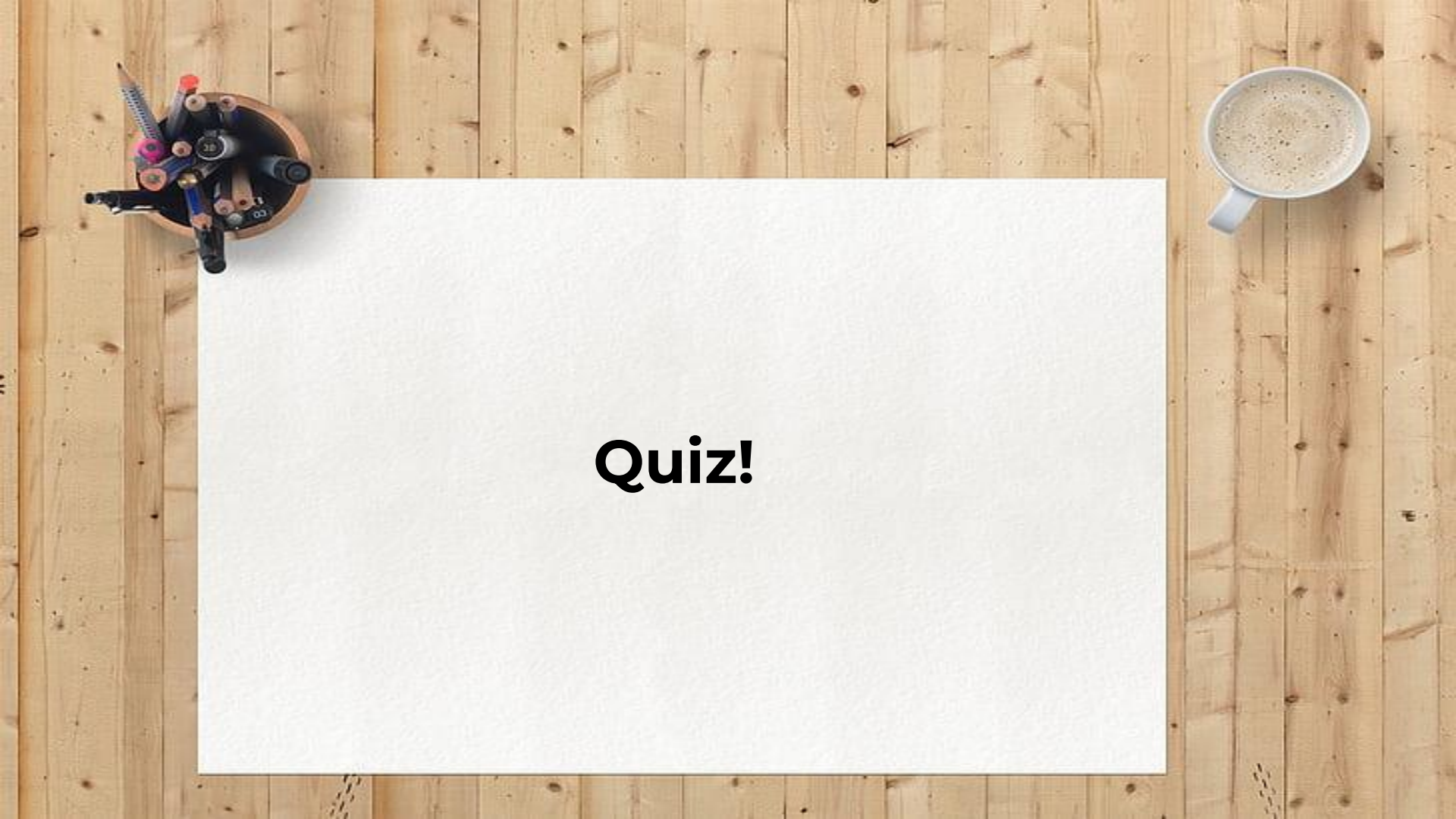
Feature	Git	Github
What is it?	Version Control System	Hosting Platform for Git Repos
Location	Primarily on your computer	Primarily Remote (cloud-based service)
Core Purpose	Tracking changes, history, branching	Hosting repos, collaboration, code sharing
Necessity	Can be used standalone	Requires Git to use
Interface	Mainly Command Line (Desktop apps exist)	Mainly web interface (Desktop app & Command Line tools exist)
Key Verbs	add, commit, branch, merge	push, pull, fork, pull request

Connecting Worlds: Why Push to GitHub?

- **Why connect your local Git repository to GitHub?**
 - Backup: Secure copy of your work off your machine
 - Accessibility: Access your code from anywhere
 - Sharing: Show your projects to others (potential employers, collaborators)
 - Collaboration: Allow others to contribute to your project
- **Once pushed, you can use a number of features on Github**
 - Pull Requests: A way to propose changes and have them reviewed
 - Issues: A system for tracking bugs, tasks, and discussions
 - Forking: Creating a copy of someone else's repository.
 - GitHub Pages: Hosting websites directly from your repository



Demo Time!

A top-down view of a wooden desk. In the center is a large white rectangular paper. In the top-left corner of the paper is a small wooden bowl filled with various colored pencils and pens. In the top-right corner of the paper is a white mug filled with a frothy beverage. The word "Quiz!" is written in the center of the white paper.

Quiz!

Question 1

- What is the primary purpose of a Version Control System (VCS)?
 - A) To write code faster
 - B) To manage changes to files and projects over time
 - C) To design user interfaces
 - D) To connect to the internet

Correct Answer: B

Question 2

- In Git, what does the term "snapshot" refer to?
 - A) A picture of what all your files look like at a specific moment
 - B) A backup copy of your entire hard drive
 - C) A list of changes made to a single file
 - D) The current version of your code in the editor

Correct Answer: A

Question 3

- Which of the following is NOT a benefit of using a Distributed Version Control System (DVCS) like Git?
 - A) Ability to work offline
 - B) Faster operations for most actions
 - C) Reliance on a central server for all operations
 - D) No single point of failure

Correct Answer: C

Question 4

- What is the purpose of the Staging Area (Index) in Git?
 - A) To permanently store the project history
 - B) To edit files directly
 - C) To display the differences between file versions
 - D) To prepare changes for the next commit

Correct Answer: D

Question 5

- Which Git command is used to save changes from the Staging Area to the Repository?
 - A) git add
 - B) git push
 - C) git commit
 - D) git save

Correct Answer: C

Question 6

- What is the purpose of branching in Git?
 - A) To create a backup of your code
 - B) To share code with others online
 - C) To work on new features or bug fixes in isolation
 - D) To delete files from the repository

Correct Answer: C

Question 7

- Which of the following best describes GitHub?
 - A) A command-line tool for version control
 - B) A platform for hosting and collaborating on Git repositories
 - C) A programming language
 - D) An operating system

Correct Answer: B

Question 8

- Which Git command is used to upload local commits to a remote repository like GitHub?
 - A) git pull
 - B) git fetch
 - C) git push
 - D) git merge

Correct Answer: C

Homework: Complete the website!

- Task: Complete the two of the remaining sections of the [travel website](#)
 - Destination
 - Testimonials
 - Newsletter

References

- [Git Documentation](#)
- [Pro Git Book](#)
- [Learn Git Branching \(Interactive\)](#)
- [Github Skills](#) (official Github introduction)
- [Github Documentation](#)