REPUBLIC POLYTECHNIC
DISCOVER. TRANSFORM. ACHIEVE

C206 L06b

Version Control using Git

SOI
SCHOOL OF INFOCOMM

L06b Learning objectives

- Understand the differences and use cases of Local, Centralized, and Distributed Version Control Systems to select the best system for project needs

- Set up and use Git and GitHub within an Integrated Development Environment (IDE) to streamline version control processes in software development

- Apply the Git workflow, including staging, committing, branching, and merging, to manage code changes effectively and collaborate in team projects

Version Control & Source Code Management

Version Control & Source Code Management

Version Control is the process of managing changes to computer programs, documents, and other collections of information, enabling tracking of individual versions and changes over time.

Source Code Management (SCM) is a broader term that encompasses Version Control as well as other aspects of managing software projects including collaboration, maintaining code integrity, and managing the steps and tools involved in the software development lifecycle from development to deployment.

Version control systems

- Version control is a system for managing **changes** to files over time
- It allows the tracking and recalling of specific file versions
- Essential for software development, particularly for **team collaborations**
- Helps **prevent conflicts** when multiple people work on the same project
- Maintains a **history** of every modification in a specialized database
- Enables **reverting** to earlier **versions** to fix mistakes or compare changes

https://www.git-tower.com/learn/git/ebook

https://www.git-tower.com/learn/git/ebook/en/desktop-gui/basics/what-is-version-control

https://www.git-tower.com/learn/git/ebook/en/desktop-gui/basics/why-use-version-control

3 types of version control systems

Local Version Control Systems

These systems keep track of file changes in a local database on the computer where the files are located.

Centralized Version Control Systems (CVCS)

In CVCS, a single server contains all the versioned files, and several clients check out files from this central place.

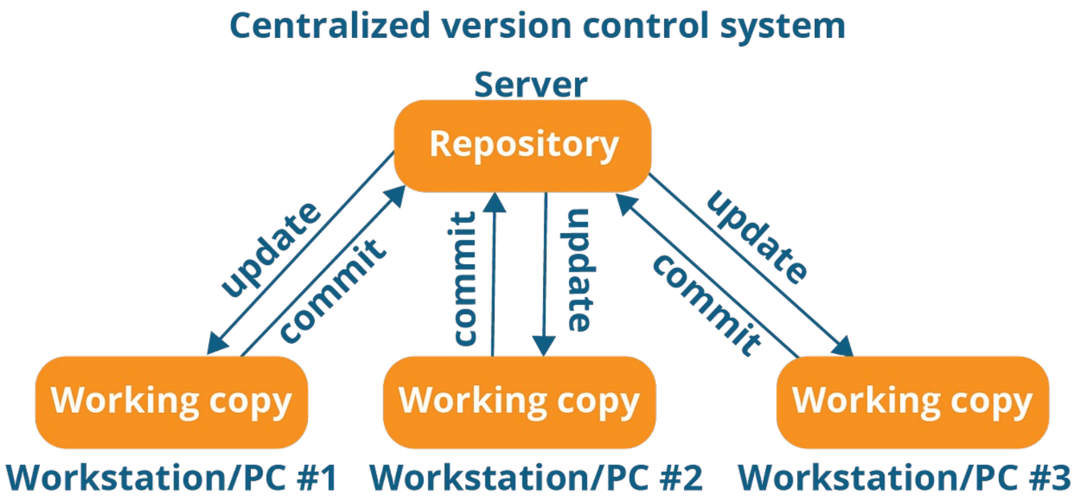
This setup is more collaborative compared to local VCS.

Distributed Version Control Systems (DVCS)

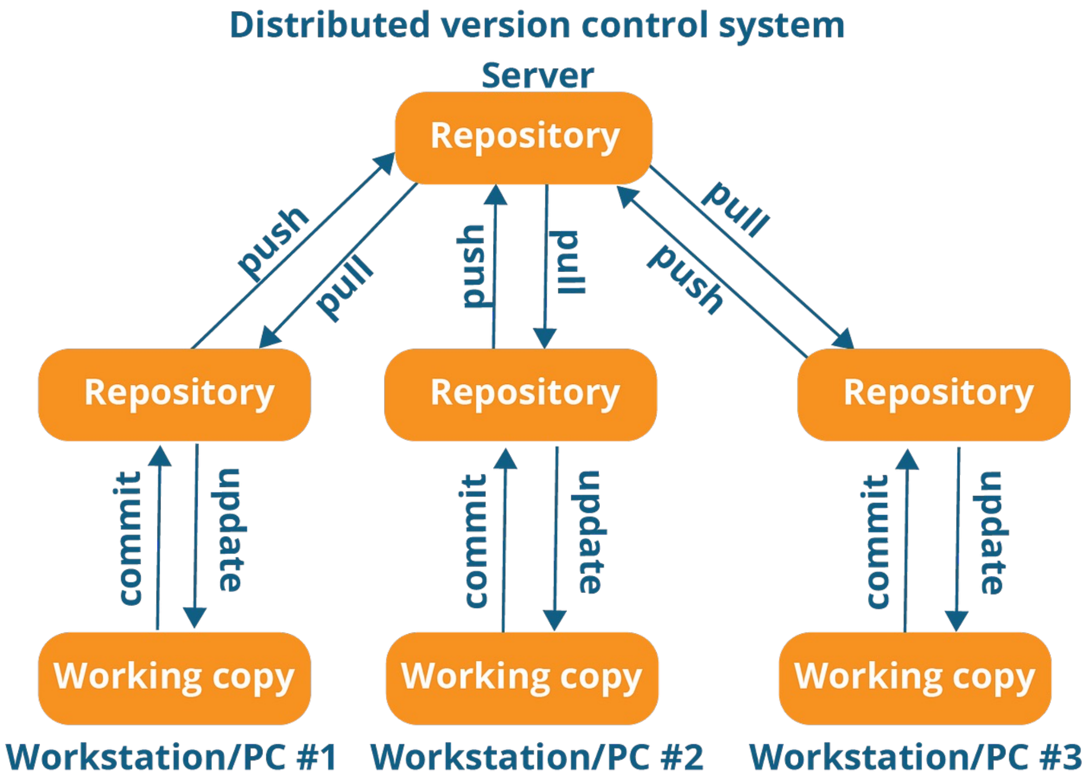
Every user's workstation effectively acts as a repository with a full history of changes.

DVCS allows for multiple mirrored repositories and enables more flexible collaboration among team members.

Centralised vs Decentralised



Centralized version control system



Distributed version control system

Centralised vs Decentralised - Comparison

Aspect	Centralized VCS (CVCS)	Decentralized VCS (DVCS)
Repository Location	Central server	Local on each user's machine
Network Dependency	High for most operations	Only for syncing changes
Risk	Single point of failure	No single point of failure
Commit Visibility	Visible to all after commit	Local until shared
Collaboration	Managed through central server	Peer-to-peer sharing
Branching & Merging	Can be complex due to centralization	Easier due to local management
Suitability	Small to medium teams	Large distributed teams / open source projects
Offline Access	Limited	Full access to history and versioning
Data Redundancy	Server only	Every clone is a full backup

Centralised vs Decentralised - Examples

Version Control System Type	Open Source Examples	Proprietary Examples
Centralized	SVN (Subversion)	Perforce
	CVS (Concurrent Versions System)	IBM Rational ClearCase
Decentralized	**Git**	BitKeeper
	Mercurial	Microsoft Team Foundation Server (before rebranding to Azure DevOps)

Git

Pro Git book - https://github.com/progit/progit2/releases/download/2.1.426/progit.pdf

Watch this video - https://www.linkedin.com/learning/learning-github/what-is-git (3m 20s)

- Git is a free and open-source distributed version control system for tracking changes in the **source code** or any other files during software development
- It needs to be installed on the local computer
- git can be run from the command line but is integrated into many IDEs, including Visual Studio Code
- For this module, we will not be using git from the command line

Version Control with Git

Git Tutorial Part 1: What is Version Control?

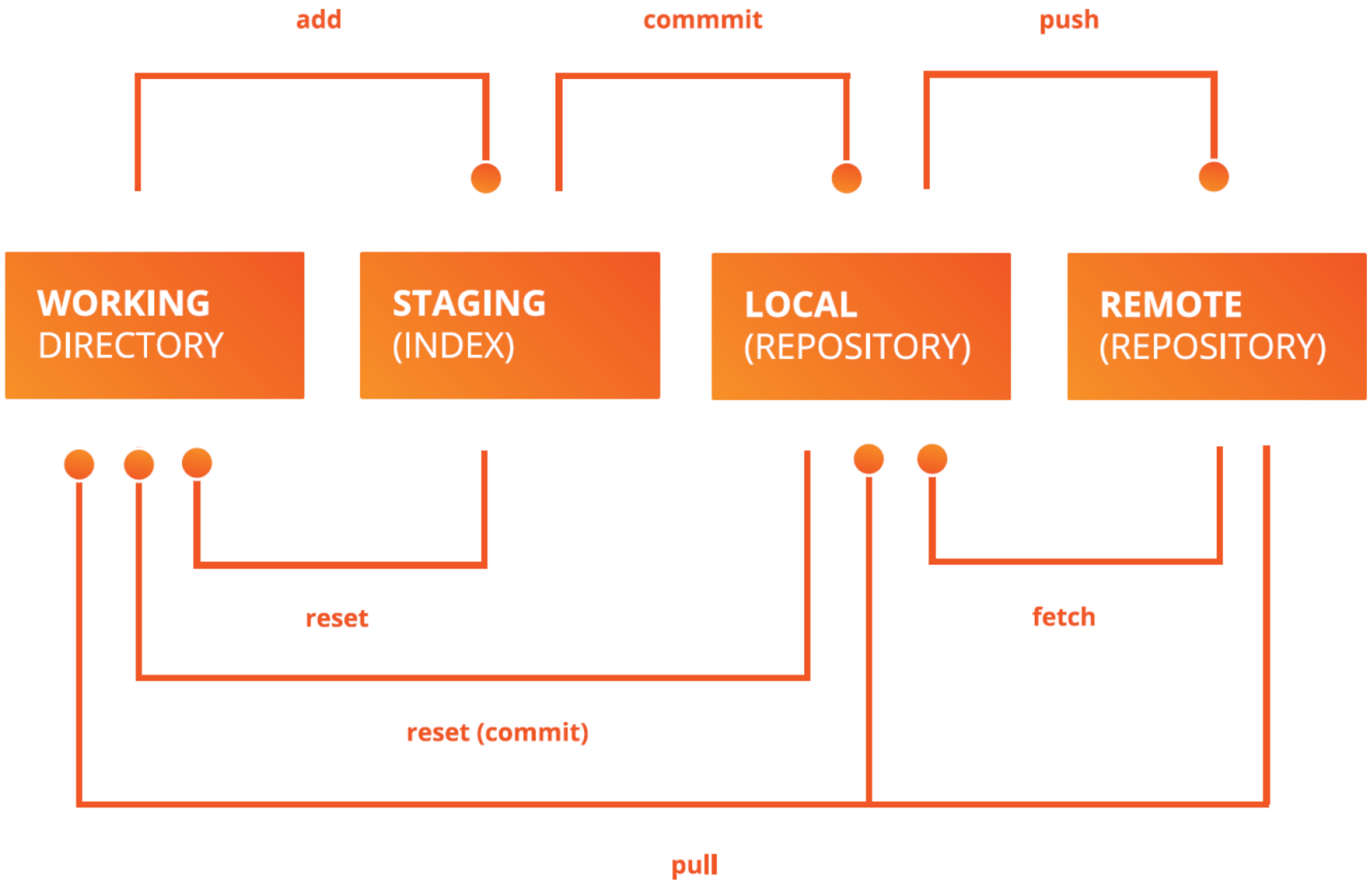
- What is Git?
- Why should we learn Git?

Watch this video - https://www.youtube.com/watch?v=9GKpbI1siow (9m 35s)

The Basic Workflow of Version Control

Watch this video - https://www.youtube.com/watch?v=e8PGuOyZ3YU (3m 4s)

Git concepts



add

commmit

push

WORKING DIRECTORY

STAGING (INDEX)

LOCAL (REPOSITORY)

REMOTE (REPOSITORY)

reset

fetch

reset (commit)

pull

Git concepts

Key Concept	**Description**
Repository	Storage space for your project, can be local or on a server
Clone	Create a local copy of a remote repository
Stage	Prepare changes for a commit by adding them to a staging area
Commit	A snapshot of your repository changes at a specific point in time
Branch	A parallel line of development, diverging from the main project
Merge	Apply changes from one branch to another
Pull	Fetch and merge changes from a remote repository into your local branch
Push	Send local repository changes to a remote repository branch
Pull Request	Propose changes to a project for review and merge into the base branch
Fork	Create a personal copy of another user's repository

GitHub

GitHub docs - https://docs.github.com/en/get-started/

Watch this video - https://www.linkedin.com/learning/learning-github/what-is-github (1m 55s)

How GitHub works
Watch this video - https://www.youtube.com/watch?v=w3jLJU7DT5E (3m 32s)

- GitHub is a web-based Git version control repository hosting service, but with many other features such as a wiki and basic task management

Best practices

Commit frequently

- Commit every time after you finish a logic unit / task

Synchronize / **update** frequently

- Keep your files up to date with the latest changes

Test before commit

- Make sure your code passes all unit tests
- Do not commit code that cannot compile as it will disrupt the teams' progress

Give meaningful commit **comments**

- Describe the changes / reasons for the commit
- Reference the task you are working on e.g. Bug ID or Task ID

.gitignore file

https://git-scm.com/docs/gitignore

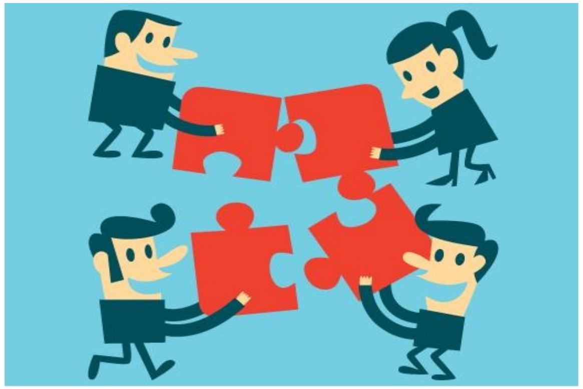
Specifies intentionally untracked files to ignore

https://github.com/github/gitignore

A collection of useful .gitignore templates

Hands on activity - TestCalculator

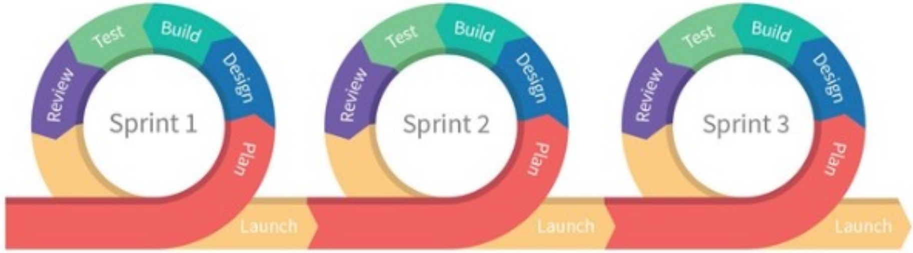
- Create a **public repository** for the Calculator and TestCalculator from the lesson on unit testing
- Copy the project web URL and paste it in the worksheet



Summary

- Understand the differences and use cases of Local, Centralized, and Distributed Version Control Systems to select the best system for project needs
- Set up and use Git and GitHub within an Integrated Development Environment (IDE) to streamline version control processes in software development
- Apply the Git workflow, including staging, committing, branching, and merging, to manage code changes effectively and collaborate in team projects

L07 Pre-reading



https://plan.io/blog/ultimate-guide-to-implementing-agile-project-management-and-scrum/