Git

Version Control System

Contents

Here's what you'll find in this.

1. What is a VCS And Why VCS ?
2. Centralize vs Distributed VCS
3. What is Git ?
4. Git repositories
5. Versioning with Git
6. Github
7. Git concepts
8. Lots Git Commands
9. Github SSH Login

Our Daily Tasks

★ Create text

★ Save text

★ Edit Text

★ Save it again

What is Version Control System

- About managing multiple versions of
  - Documents
  - Programs
  - Websites etc
- Tracks History of collection of files
- **Version control** software keeps track of every modification to the code in a special kind of database

Why VCS ?

★ **For Individual Help:**

  ○ Gives you a “time machine” for going back to earlier versions

  ○ Gives you great support for different versions (standalone, web app, etc.) of the same basic project

★ **For Working with Team:**

  ○ Greatly simplifies concurrent work, merging changes

★ **Management of changes to files.**

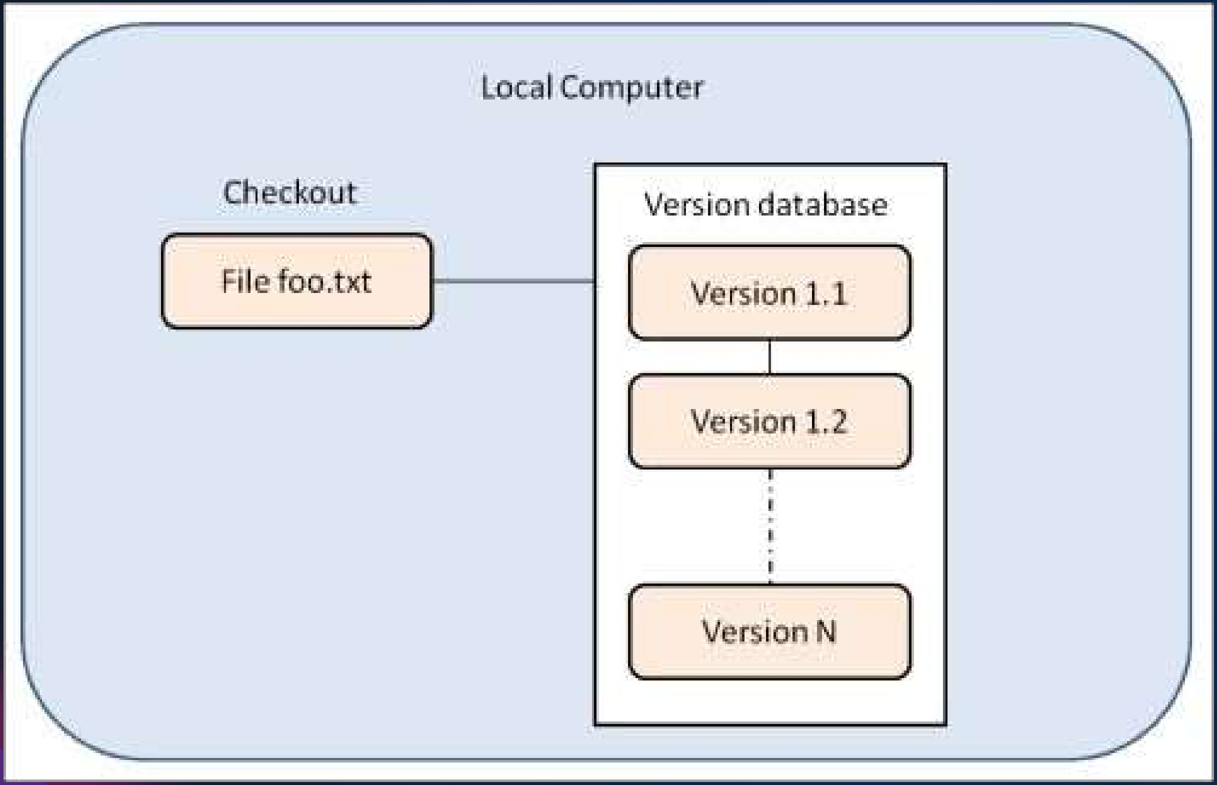
  ○ Keep track of what changes occurred.

  ○ Allows People to work Together.

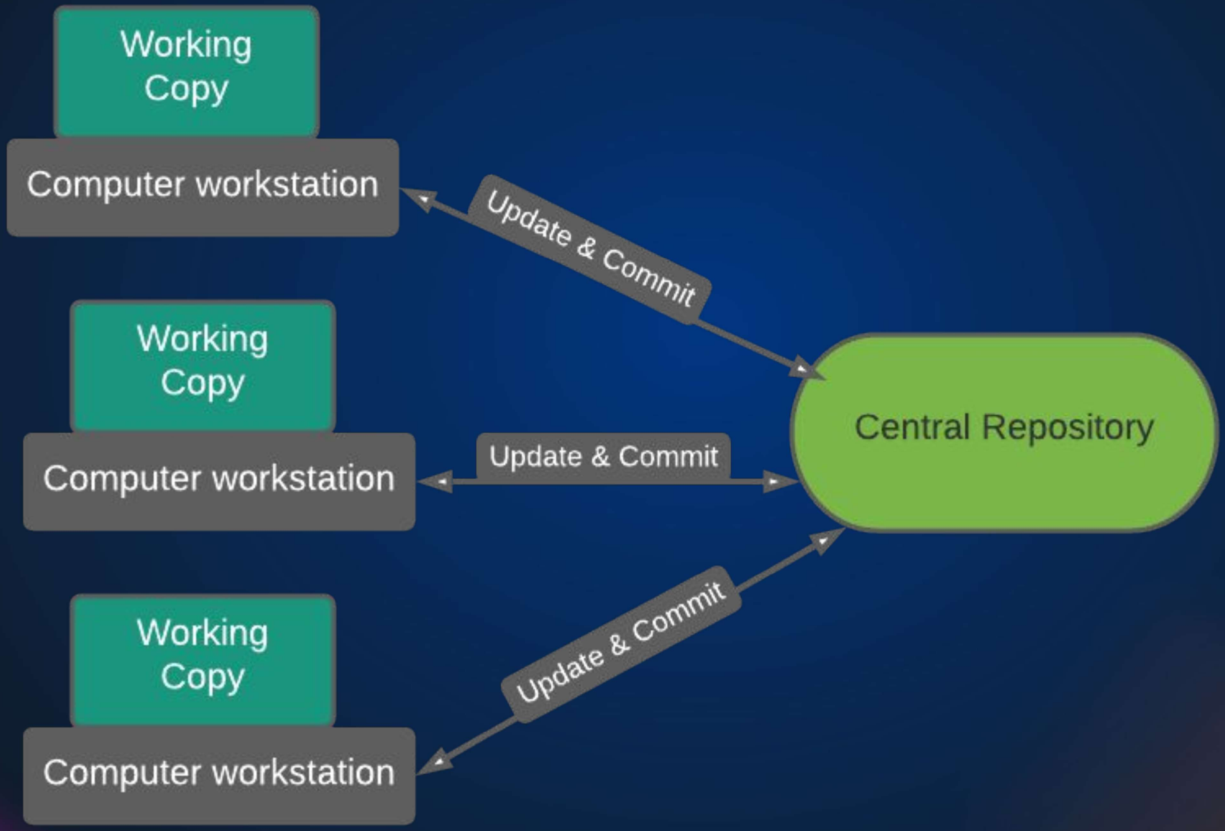
Localized and Centralized VCS

- A localized version control system keeps local copies of the files.
- In centralized source control, there is a server and a client. The server is the master repository which contains all of the versions of the code.

Localised VCS



Centralized VCS



Centralized VCS

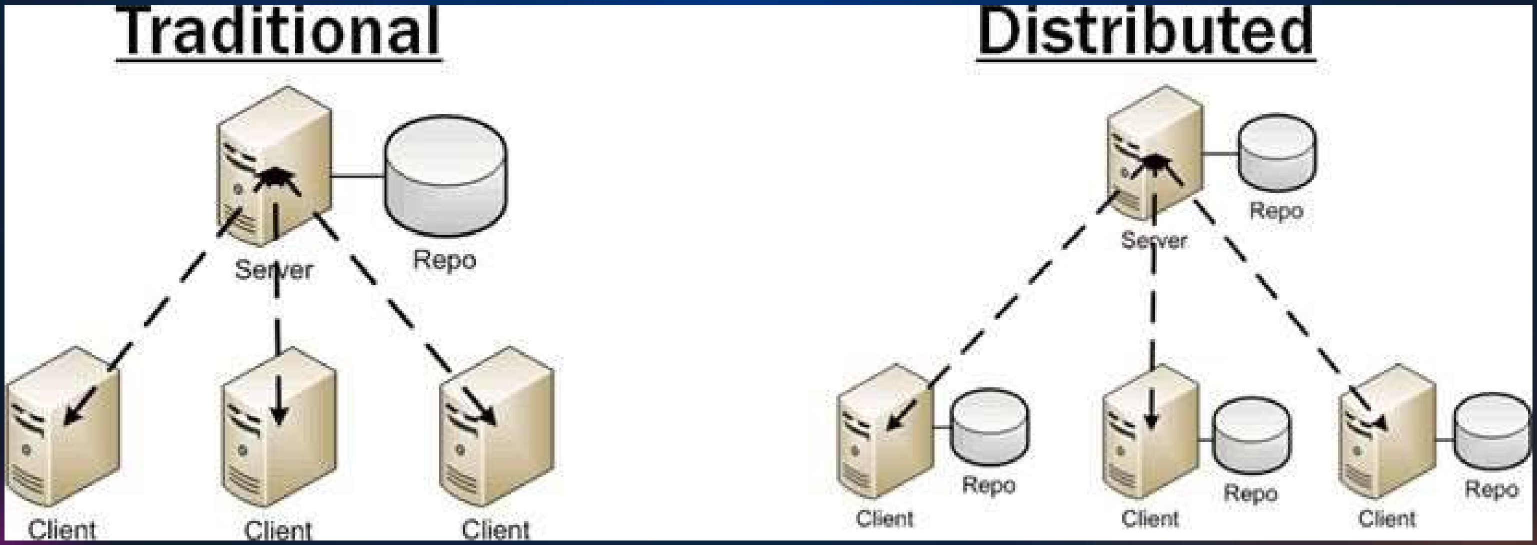
- In Subversion, CVS, Perforce, etc. A central server repository (repo) holds the "official copy" of the code.
  - the server maintains the sole version history of the repo
- You make "checkouts" of it to your local copy
  - you make local modifications
  - your changes are not versioned
- When you're done, you "check in" back to the server
  - your checkin increments the repo's version

Drawbacks

- Both approaches have the drawback that they have one single point of failure.
- In a localized version control systems it is the individual computer and
- In a centralized version control systems it is the server machine. Both system makes it also harder to work in parallel on different features. Eg:Git,mercurial etc.

Distributed version control systems

- In a distributed version control system each user has a complete local copy of a repository on his individual computer.



Distributed version control systems

In git, mercurial, etc., you don't "checkout" from a central repo

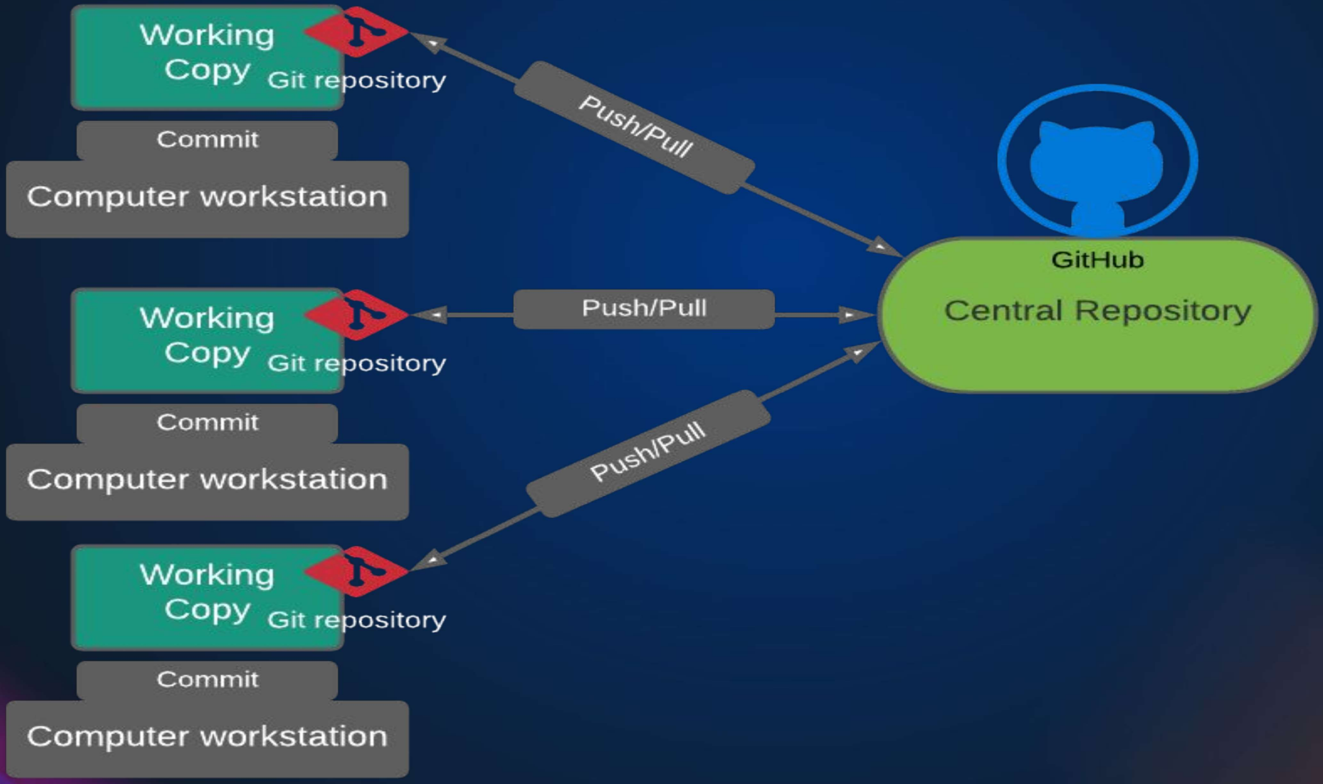
  - you "clone" it and "pull" changes from it

- Your local repo is a complete copy of everything on the remote server
  - yours is "just as good" as theirs

- Many operations are local:
  - check in/out from local repo
  - commit changes to local repo
  - local repo keeps version history

- When you're ready, you can "push" changes back to server

Distributed version control systems



What is Git ?

- Git is a distributed version control system
- Git is a Tree History storage system
- Git is content tracking management system

Git Provides

Speed

- Support for non-linear development
- Fully distributed
- Able to handle large projects

Ease

Simple to use tools & commands.
Cloud based remote repository.

Git Creator

- Created by Linus Torvalds, creator of Linux, in 2005
  - Came out of Linux development community
  - Designed to do version control on Linux kernel

Installing Git

Windows

Git Bash

Mac OS

Install via HomeBrew

Linux

Install via Package manager
(yum, apt, snap etc)

Local Repository Setup

1. **Set the name and email for Git to use when you commit:**
   - ➢ git config --global user.name "Imran Teli"
   - ➢ git config --global user.email imran@visualpath.com

2. **Create a directory**

3. **Initialize dir with**
   - ➢ git init

4. **Create Readme.md file**
   - ➢ git add (Staging)
   - ➢ git commit (Local commit)

Remote Repository

- Create Remote repository on
  - Github, bitbucket, codecommit etc

- Clone Repo to local
  - git clone URL

- Local to Remote integration
  - cd to local repo
  - git remote add origin ssh://git@github.com/[username]/[repository-name].git
  - git push
  - git pull (to fetch latest changes)