Git-Day-1

15 November 2023 09:48

- **What is a version control system?**
  Version control system also referred as **source code management**

- In simple words
  Version control system we can call as **VCS &**
  Source code management tool has **SCM** tool

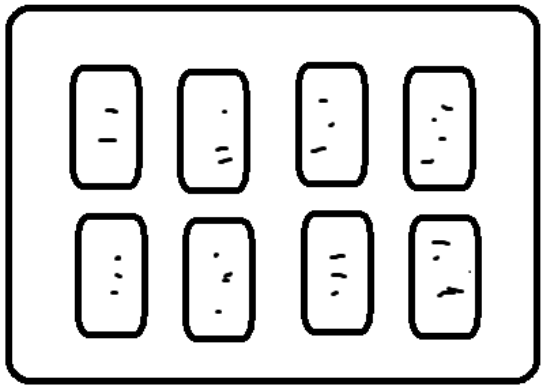
- **Different type of VCS tools** in market,
  1. GIT
  2. SVN
  3. Clear case
  4. Mercurial

- Before we understand what is an version control system? we will discuss **what are all the problems that will occur them version control system**.

  - **Scenario:** Just assume you are developer and working in MNC IT company & your manager came to you and told that we have new project from some client and he gave you some inputs to start work on that project.

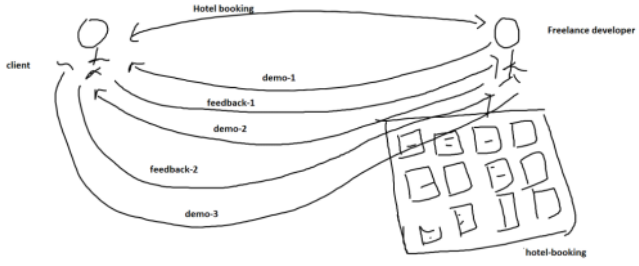
  - So as a next step what you will do based on the inputs that you received from the manager and client you started working on developing of application.

  - Developer will create a folder in his laptop & will write the code in the files.



railway-project

  - Once the developer completed first version(v1) of project, developer will set up a meeting with client & manager for reviewing application.



  - During the first version of application review the manager & client are not satisfied with the futures that are present in application. So they suggested few more corrections to your application & new features.

  - So as a next step the developer what will do?
    Again he will go back to his system and based on the inputs that received from the review meeting he will correct the source code and add new other features, correct or not?

  - After the completion of the code corrections & futures development based on the review-1 meeting and again you are going to set one more review meeting with client and manager to validate V2 code.

  - During the review-2 meeting also there are some more changes suggested by Manager & Client.

  - So again developer what he will do?
    He will update the code in existing folder based on the review-2 meeting inputs.

  - Again developer will setup review-3 meeting with manager & Client to review V3 application & its code.
    In this review-3 meeting manager & client said we don't need these changes latest but i want features that presented during the review-1 meeting that are fine.

  - **Now it's a big trouble to developer, he can't go back to the exact v1 files of code, correct or not?** correct

  - **Why correct?**

There are so many lines of code changed & many files are added, so he can't go back to exactly v1 code.
V3 ---> V2 ----> V1

- **Problems without VCS**
  - Overwritten the files & not having mechanism to rollback to previous changes on files.
  - Assume when multiple developers are working on same project, it will be very hard to find who is working on what & collaborat ion will be very difficult.

- **VCS will resolve all these kind of issues like,**
  - Record the changes happened to the files
    - What is changed
    - When it is changed
    - Who is changed
  - Allows to rollback to previous versions of files
  - Allows to review the latest source code files with previous versions of files.
  - Acts as a best collaborative tool for developers, so it's easy to know who is working on what & maintain stable code.

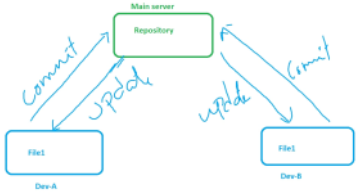
- Types of VCS
  - CVCS - Centralized Version Control system
  - DVCS - Distributed Version Control system

- **Centralized version control system**
  - In centralized version control system we will have one central server & which contains repository.

  - **What is an repository?**
    Repository is nothing **folder which is contains source code files** & also the **metadata about the files**.

  - **What is metadata about files?**
    Who is modified the file, when is it modified & who modified like this information stored.

  - Now let's see how CVCS will work

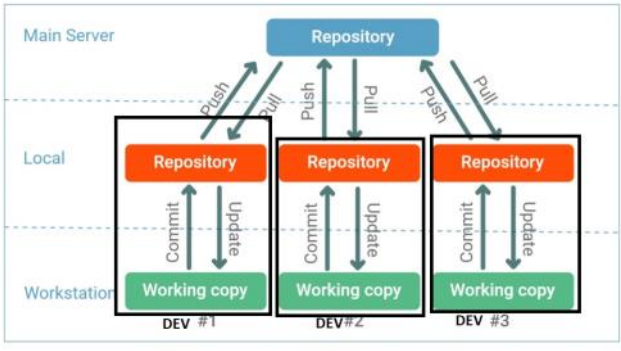


Assume there are two are developers are present developer-1 & developer-2 & these two developers are connected to centralized repository
- Developer-1 connected to the central repository & did checkout of files that he need to update.
- Now the changes made by the developer-1 present in laptop only.
- In order to make the changes visible to others the developer-1 should commit those changes to central server.
- Now developer-2 able to see the changes made by developer-1.

  - **Problems in CVCS:**
    - Single central repository always developers are connecting, so if there is any network issues developers will sit idle.
    - If central server goes down that's it we lost the code, if we don't have proper backup mechanism.

  - **SVN & IBM-Clear case comes under CVCS category**

- **Distributed version control system(DVCS)**



  - In DVCS the **main server contains repository** & this repository we can call it also **remote repository**.
  - Developers will **clone the copy of remote repository into laptop** & this repository called as **local repository**.
  - On this local repository developers user will perform update/commit operations to make changes reflect.
  - In order to make the changes visible to others developers can push changes to remote repository.

  - **Benefits of DVCS are CVCS**
    - If main server goes down & we can recover the repo from latest copy local repository.
    - Even if there is network issue developers can work on local repositories & can push the changes to main repo once network ava ilable.

  - **Git comes under DVCS category.**

==========

- **How to install Git on Windows**

==========

  - Download .MSI way
  - Using choco
    - **Install choco software in windows**
      Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))

    - **Install git**
      choco install git

    - **Check git is installed or not**
      git --version

==========

- **How to install Git on Linux**

==========

yum install git -y
(or)
dnf install git -y

==========

Git config

==========

- This command will help us to introduce ourself to Git like who is using person using it & his mail id.

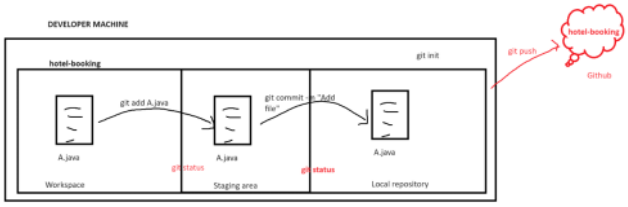
- git config --global user.name chaitanya
  git config --global user.email chaitanya.rv99@gmail.com

- To check this configuration we will run command
  git config --list

==========

Create a Local repository & Explain different stages

==========



- Imagine this is a **developer-1** machine with **windows OS** & git already installed on it.

- On this developer machine a folder called hotel-booking created by developer-1 & this directory will have source code files related to the project will be developed.

- To **convert this hotel-booking directory as a repository** we have to run "**git init**" command inside the **hotel-booking** directory.

- After initialization of the **hotel-booking** as a repository this **hotel-booking** directory will be divided into three sections logically.
  - **Working directory**
  - **Staging Area**
  - **Local repository**

- Now the developer started working on the project and he created **file A.java in the working directory** and he written some code inside A.java file.

- To move the file from **working directory to staging area** we can do it by running the add command.
  git add A.java

- In order to make sure whether the file is staging area are in the work space we can run the **git status** command.
  From this command we will come to know whether the file is in which stage.
  So now we can see the file is in staging area.

- To move the file from **staging area to local repository** you can use the command **commit**.

- When we run commit command in the repository the snapshot of the changes will be taken and committed .
  Along with that information you will get like
  - who committed those changes
  - when those changes are committed and
  - commit message.

- **Explain this scenario practically**

- To check what commits present on the repository we can run a command **git log**.

- Here we can see the first comment we made on the repository.
  - 40-bit SHA code
  - Mail-id of the user
  - User ID
  - Commit message.
- Now the changes are present in developer machine only & in order push the local repository into internet & make it visible to other developers we have to use repository hosting platforms.

- GitHub
- Azure devops
- Bit bucket
- GitLab

==========

➢ **Create GitHub account & empty remote repository**

==========

- Now I will show you how to create a remote repository.
- Where do we create our remote repositories? GitHub/Azure devops/Gitlab
- First of all what is **GitHub**?
  GitHub is the repository hosting platform, it means it will store all the repositories and make those repositories available to the other developers for easily collaboration.
- **People might think git and GitHub both same. Is this correct?**
  - No it's not correct statement.
  - **Git** is a **tool** &
    GitHub is a .**repository hosting platform**
  - Like camera is a tool and Facebook or Google drive or Google photos is a hosting platform like GitHub.
- **Now let's see how to create GitHub account.**
  - To create GitHub account you need an email id so since I already have the GitHub account I am just giving the random email ID .
  - After creating the GitHub account the email id will receive an 8-digit passcode and we have to enter this password then only our account creation will get complete.
- **Now let's understand what is an project?**
  - Under project we can create the number of a repositories that belongs to specific project.
  - Suppose hospitals is my project then the different departments like
    oncology
    radiology
    cardiology
    gastrology
    and all those things we can consider as repositories.
  - Now we are skipping the part of creating the organisation as of now and we are going to create repo under the default organiz ation & default organization name userid name.
- **Create remote repository in GitHub with same name as local repository.**
  - To create repository under the default organisation we need to pass unique repository name and
  - Give the description about the repository the description is an optional thing, but we can provide it.
  - Next we need to choose repository type as a public or private.
  - **so what do you mean by public & private repository?**
    - **Public** repository can be **accessible over the internet by any person** &
    - If repository **private** s a only the **specific people that you granted access**.
- Next don't choose the checkbox like readme.md file & create **empty remote repository**.
- Now we have an empty remote repository in GitHub & local repository in developer-1 machine. In order to push changes from local repository to remote repository connectivity required between them.
- First let's **check is there any connectivity between local repository & remote repository**
  git remote -v
  There is no connectivity b/w local & remote repo's.
- To setup connectivity between local & remote repo we need to do
  git remote add origin <github_url >

  git remote -v
- Now developer-1 is ready to push the code.
  git push origin main
- GitHub repo is now updated with local repository changes.