GIT

Open Source Distributed Version Control System

What is Git?

Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency

What is version control

Version control, also known as source control, is the practice of tracking and managing changes to software code. Version control systems are software tools that help software teams manage changes to source code over time.

Types of version control

Version control systems (VCS) are tools that help manage changes to source code or other collections of information. They are critical for collaboration, maintaining a history of changes, and ensuring project integrity. There are several types of version control systems, which can be broadly classified into three main categories: local, centralized, and distributed.

1. Local Version Control Systems
Local version control systems are the simplest form of VCS. They keep all versions of the files on the local machine. This type of VCS is useful for small projects or individual use but lacks collaboration capabilities.

- **Example**: RCS (Revision Control System)
  - RCS stores each change as a set of differences (deltas) from a base version.
  - It works by maintaining a set of patch files that can recreate any previous version.

2. Centralized Version Control Systems (CVCS)
Centralized VCS have a single central server that contains all the versioned files. Clients check out files from this central place. CVCS allows multiple users to collaborate on a project, but the central server can be a single point of failure.

- **Examples**:
  - **CVS (Concurrent Versions System)**
  - One of the oldest VCS, allows multiple developers to work simultaneously.
  - **Subversion (SVN)**
  - An improvement over CVS, supporting atomic commits and better handling of binary files.
  - **Perforce**
  - Known for handling large files and scalability.

3. Distributed Version Control Systems (DVCS)
Distributed VCS have multiple repositories that are fully self-contained. Every collaborator has a full copy of the project history, allowing them to work independently and merge changes later. This model improves redundancy and offline access.

- **Examples**:
  - **Git**
  - Widely used, especially for open source projects. Offers powerful branching and merging capabilities.
  - **Mercurial**
  - Designed to be fast and easy to use, supports large projects with many contributors.
  - **Bazaar**
  - Focuses on usability and flexibility, allowing both centralized and decentralized workflows.

What is distributed version control ?

What is meant by distributed version control? A distributed version control system (DVCS) is a type of version control where the complete codebase — including its full version history — is mirrored on every developer's computer.

Why use Git ?

Git is the most commonly used version control system. Git tracks the changes you make to files, so you have a record of what has been done, and you can revert to specific versions should you ever need to. Git also makes collaboration easier, allowing changes by multiple people to all be merged into one source.

What is local & remote repositories ?

The local repository is a Git repository that is stored on your computer.
The remote repository is a Git repository that is stored on some remote computer (a server).
The remote repository is usually used by teams as a central repository into which everyone pushes the changes from his local repository and from which everyone pulls changes to his local repository.

GIT version

To check the current version of your git , type in the following command :

```
 git --version
```

Configuring git for first use

```
git config --global user.name "your name"
git config --global user.email "name@gmail.com"
```

Create a new local git repository

To create a new local git repository , use the following command :

```
git init
```

Staging and Committing the code

When you modify your files , you need to add them to the staging area in git. Which is used by git to track the changes of the files and directories . Then we commit the changes to the local repository .

Add files / directories to the staging area

To add files / directories to the staging area , we use the following command :

```
git add file1
```

Committing the changes to the repository

To commit the changes to the repo, we use the following command :

```
git commit -m “your message”
```

Git Status

Generally, We use git status to find out information about the modified files , and the files / directories inside the staging area.

Type the following command :

```
git status
```

Git log

You use git log to print out the commits history in this repository .
The log shows the author of each commit, the date of the commit, and the commit message.

Type in the following command :

```
git log
# for displaying in short
git log --oneline
```

Going back to a previous version (commit)

```
git reset --hard ff93cc4
```

Git Branches

Until now we have not manually created any branch . By default git create the master (or main) branch . Which is what we've been working with .
We use branches to support multiple parallel developments.

To create a new branch :

```
git branch newBranch
```

To switch into the newly created branch , we use the following command:

```
git switch newBranch
```

Merging branches

After we work on another branch , usually we need to merge the branch into the main branch . To do this , we use the following command :

- First we switch to the master branch

```
git switch master
```

- Now, type in the following to merge :

```
git merge newBranch
```

After running those , all the changes in the newBranch will be merged into the master branch.

Remote Git repository

When working with a team , companies use a remote git repository to manage the development . Usually they use a version control hosting provider such as github.com

This makes it possible for you and the team to collaborate remotely .

Remote repository cloning

To clone an existing remote repository to your computer , use the following command :
note : the repository url http://.....git

```
  git clone [repository url]
```

Pushing changes to a remote repository

After committing your changes , you'll need to push the changes to the remote repository . To do this we use the following command :

```
git push
```

If you have multiple branches and want to specify the branch

```
git push origin master
```

If you have multiple branches and want to push all changes across all branches

```
git push --all
```

Pulling changes from remote repository

When you need to pull the changes done to the repository by other developers , you use the following command :

```
git pull
```

If you have multiple branches and want to specify the branch

```
git pull origin master
```

If you have multiple branches and want to push all changes across all branches

```
git pull --all
```

Add a remote origin to your git

To add a remote origin to your local git , type in the following command :

```
git remote add origin [repository url]
```

Change the remote origin of your git

To change the remote origin to your local git , type in the following command :

```
git remote set-url origin [repository url]
```