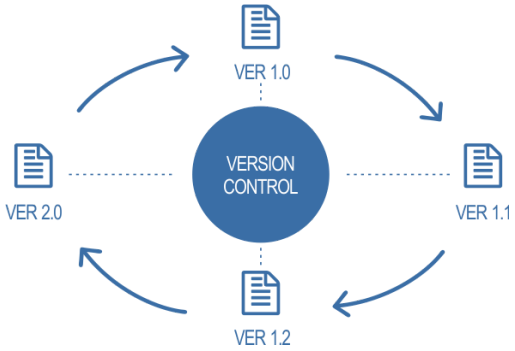
# Introduction to Version Control - GitHub Repository

## What is Version Control?

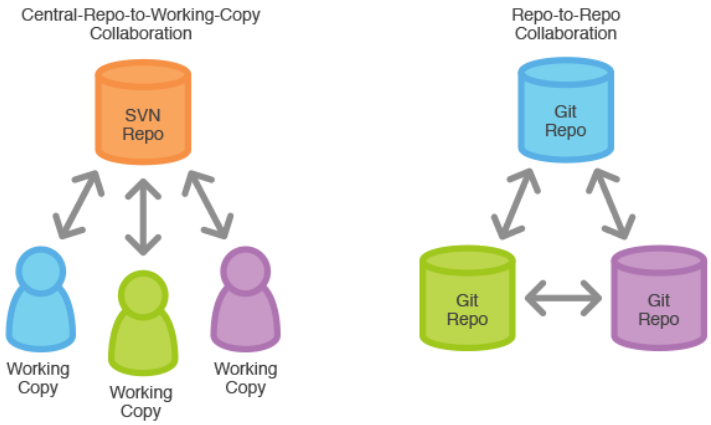
A version control system ( VCS ) is a software that keeps records of changes on each file and folders over the time. You can segregate to identify every modification and recall these later times. There are two remarkable things a VCS allows you to do:

- Track changes you make to your files
- It enhances collaboration by simplifying working on projects with multiple people and teams.




## Types of Version Control

- ☐ Local Version Control Systems
- ☐ Centralized Version Control Systems
- ☐ Distributed Version Control Systems

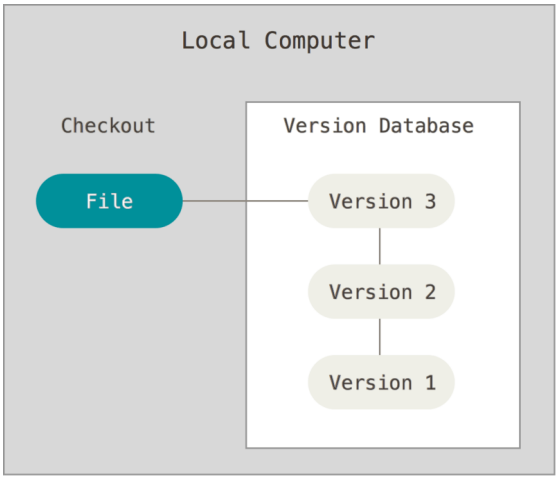



## What is SVN (Subversion)?

SVN stands for **Subversion**. It is called SVN because of its commands (its command name svn). It is a **centralized version control system &** an **open-source** tool.

## What is Git?

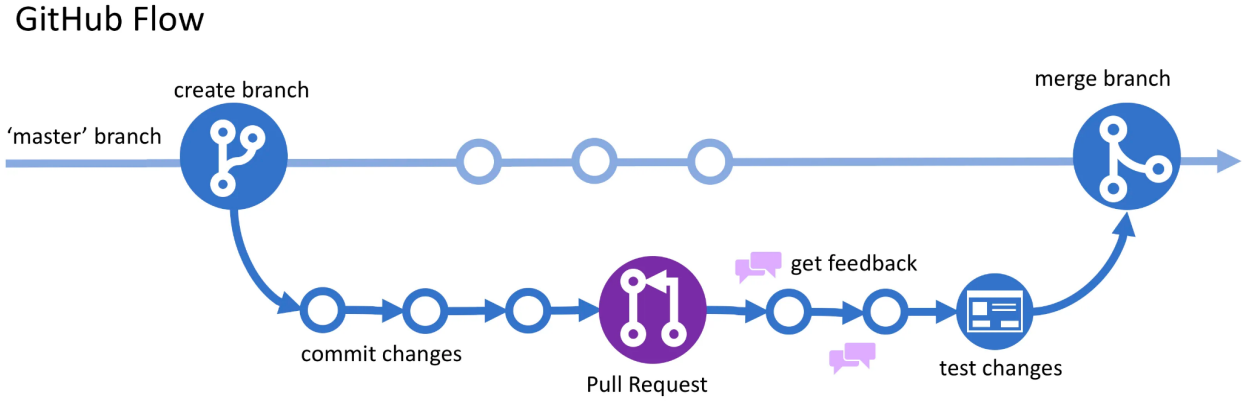
This is a source code version control software which we always need to use for software development as well as deployment. So this is a very important tool to practice DevOps Culture. You need git tools to automate build configuration and maintain release updates.




Version control (git) helps to publish latest changes on each build integration which can be used to check back overall changes of the current release product.

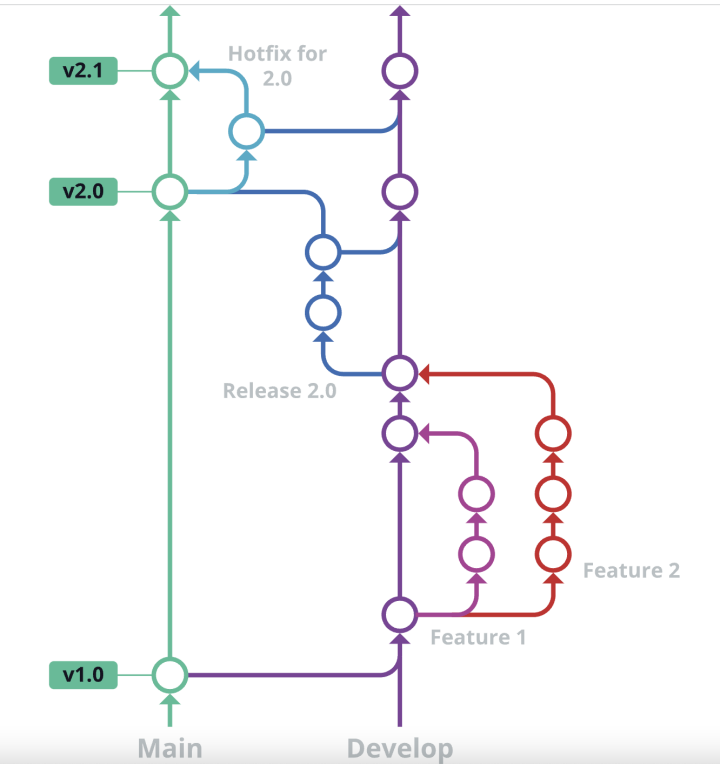
## What is a Git workflow?

Accomplishing work in a consistent and productive manner using a git recipe or recommendation is called a git workflow. It encourages developers & DevOps teams to leverage Git effectively and consistently. It offer a flexible change management of source code among teams and maintains all release.



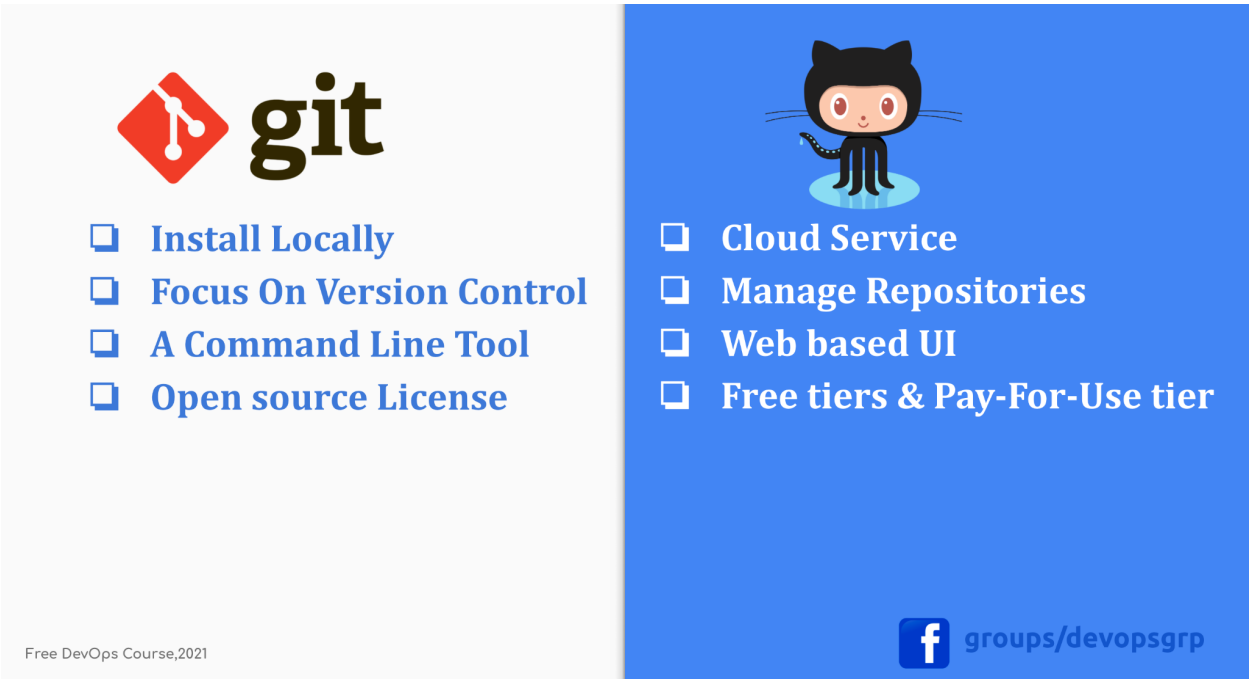
In the Git workflow , Branches are categorized into five different types.

1. Main
2. Develop
3. Feature
4. Release
5. HotFix




**Git vs GitHub**

Git lets you manage & keep track of your source code version history whereare github provides you cloud based hosting services and let s you manage your source code repositories.




I am not going to describe the entire git rules & workflow. I only focus on these terms and commands that are mostly needed to serve automatic build pipeline configure and maintain release updates.

**Benefits of Version Control System**

To write high performance software and DevOps terms version control is used as a best practice. It helps developers to move faster and allow softwarePlease terms to preserve efficiency and agility to scale up the team.

- ☐ Keep track long term history for each and every files and folders changes
- ☐ Allowing agility of development & error fixing concurrently.
- ☐ It provides the traceability of every change and helps to find errors easily to fix quickly.

## GIT Roles in DevOps

Git as a DevOps tools empower collaboration and faster release cycle. Without the git automation pipeline it is nearly impossible to deploy all changes and track the change logs during deployment and release management.

Anyone willing to start or approach DevOps practice git comes and needs to be considered as a number 1 tool which has to be considered first.

## Some Basic Commands in Git

- [ ] Create Repositories
  git init
- [ ] Make Changes
  add
  commit
  status
- [ ] Parallel Development
  branch
  merge
- [ ] Sync Repositories
  push
  pull
  add origin

## Managing deploy keys

There are multiple ways you can manage SSH key to automate deployment

- [ ] SSH agent forwarding
- [ ] HTTPS with OAuth tokens
- [ ] Machine Users
- [ ] Deploy keys

To configure CICD Pipeline we will use SSH deploy key which is more secure & reliable


During the launch project from a repository on GitHub.com to the automation server, deployed SSH keys grant access to a single repository. GitHub attached the public part and the deployment server holds the private part of the ssh key.

## Git Cheat Sheet

- [ ] https://training.github.com/downloads/github-git-cheat-sheet.pdf
- [ ] https://education.github.com/git-cheat-sheet-education.pdf
- [ ] https://about.gitlab.com/images/press/git-cheat-sheet.pdf
- [ ] https://gitcheatsheet.org/search

## Managing Deploy Key

- [ ] https://docs.github.com/en/developers/overview/managing-deploy-keys