DevOps Tools

GIT and GIT Hub

What is a Version Control System?

- Version control systems are a category of software tools that help a software team manage changes to source code over time
- Version control software keeps track of every modification to the code in a special kind of database. If a mistake is made, developers can turn back the clock and compare earlier versions of the code to help fix the mistake while minimizing disruption to all team members.

What is a Version Control System?

- For almost all software projects, the source code is like the crown jewels - a precious asset whose value must be protected.
- For most software teams, the source code is a repository of the invaluable knowledge and understanding about the problem domain that the developers have collected and refined through careful effort.
- Version control protects source code from both catastrophe and the casual degradation of human error and unintended consequences

What is a Version Control System?

- Software developers working in teams are continually writing new source code and changing existing source code.
- The code for a project, app or software component is typically organized in a folder structure or "file tree".
- One developer on the team may be working on a new feature while another developer fixes an unrelated bug by changing code, each developer may make their changes in several parts of the file tree.

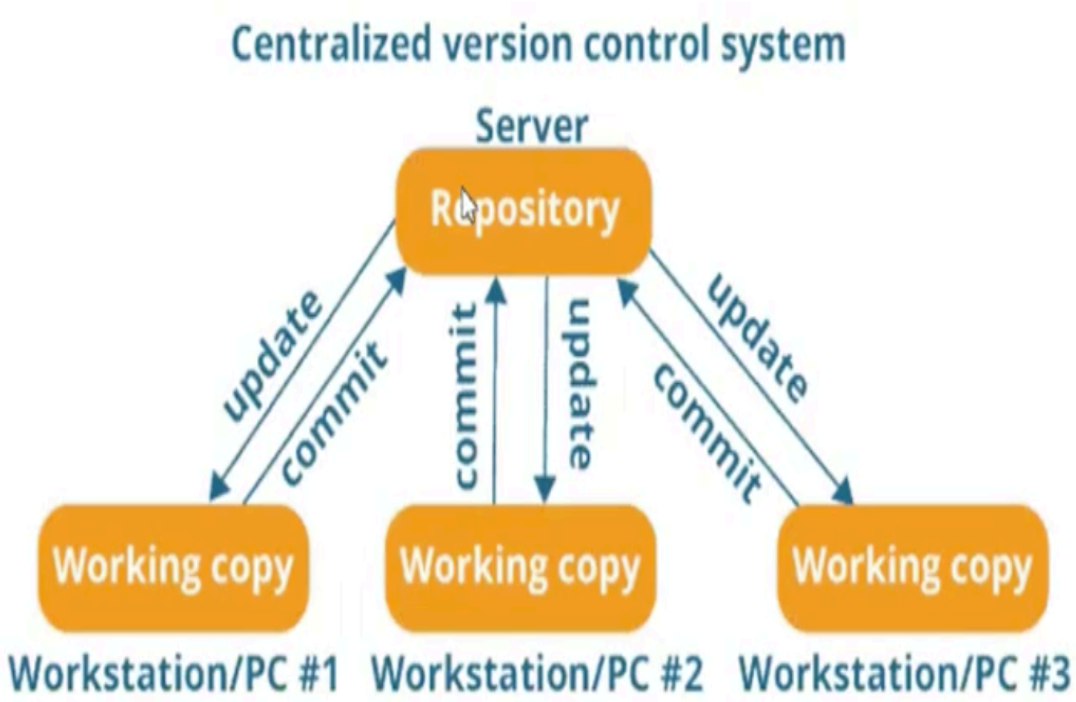
What is a Version Control System?

- Version control helps teams solve these kinds of problems, tracking every individual change by each contributor and helping prevent concurrent work from conflicting.
- Changes made in one part of the software can be incompatible with those made by another developer working at the same time.
- This problem should be discovered and solved in an orderly manner without blocking the work of the rest of the team. Further, in all software development, any change can introduce new bugs on its own and new software can't be trusted until it's tested.
- So testing and development proceed together until a new version is ready.

What is a Version Control System?

- Version control is the ability to understand the various changes that happened to the code over time (and possibly roll back).
- All these are enabled by using a version control system such as Git
- A Git repository can live on one's machine, but it usually lives on a central online repository
- Benefits are:
  - Collaborate with other developers
  - Make sure the code is backed-up somewhere
  - Make sure it's fully viewable and auditable

Types of Version Control System

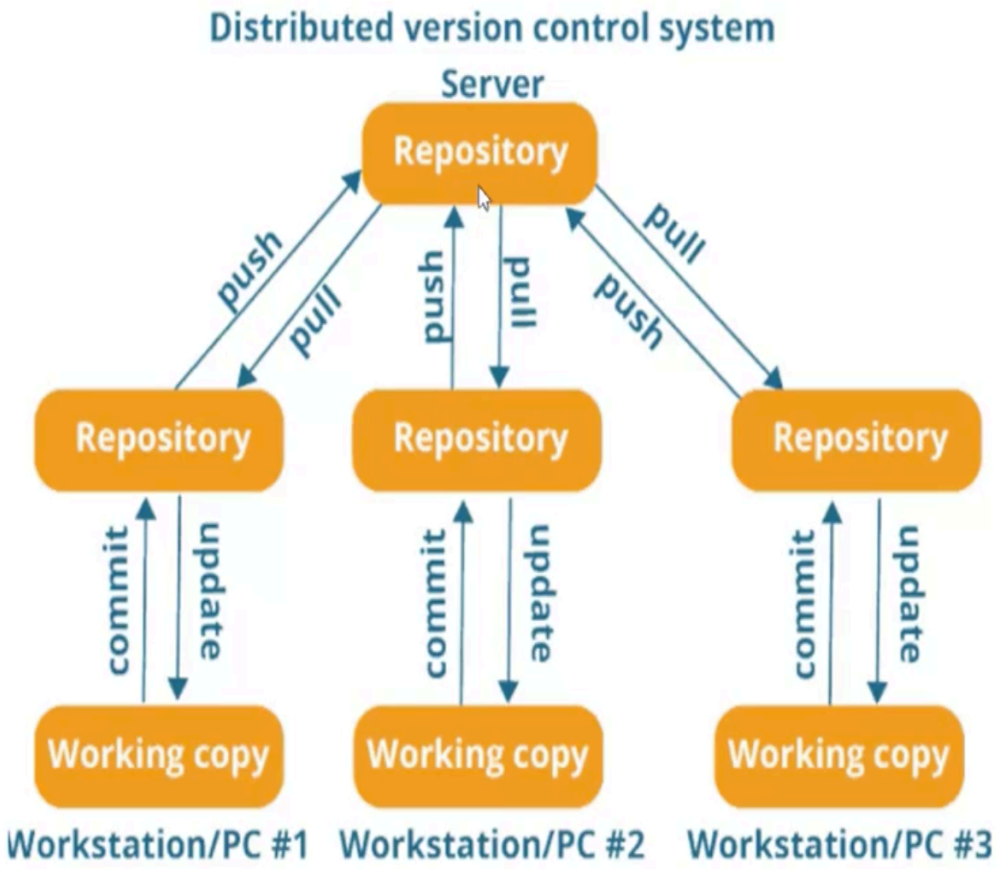


- It uses a central server to store all files
- It works on a single repository to which users can directly access
- Central server is located on a remote machine which is directly connected to each of the programmer's computer

Drawbacks of Centralized Version Control System

- It is not locally available
- Since everything is centrally available, if the central server is crashed or corrupted entire data of the project will be lost

Types of Version Control System



- In a distributed VCS,every developer has a local copy of the main repository
- Developers can update their local repositories with the new data from the central server by an operation called “Pull” and commit any local changes to the central server by an operation called “Push” from their local repository

What is Git?

- By far, the most widely used modern version control system in the world today is Git.
- Git is a mature, actively maintained open source project originally developed in 2005 by Linus Torvalds, the famous creator of the Linux operating system kernel.
- A staggering number of software projects rely on Git for version control, including commercial projects as well as open source.

What is Git?

- Having a distributed architecture, Git is an example of a DVCS (hence Distributed Version Control System).
- Rather than have only one single place for the full version history of the software as is common in once-popular version control systems like CVS or Subversion (also known as SVN), in Git, every developer's working copy of the code is also a repository that can contain the full history of all changes.
- In addition to being distributed, Git has been designed with performance, security and flexibility in mind

Why Git?

- Git is the best choice for most software teams today. While every team is different and should do their own analysis, here are the main reasons why version control with Git is preferred over alternatives
- **Git is good**
  - Git has the functionality, performance, security and flexibility that most teams and individual developers need. In side-by-side comparisons with most other alternatives, many teams find that Git is very favorable.

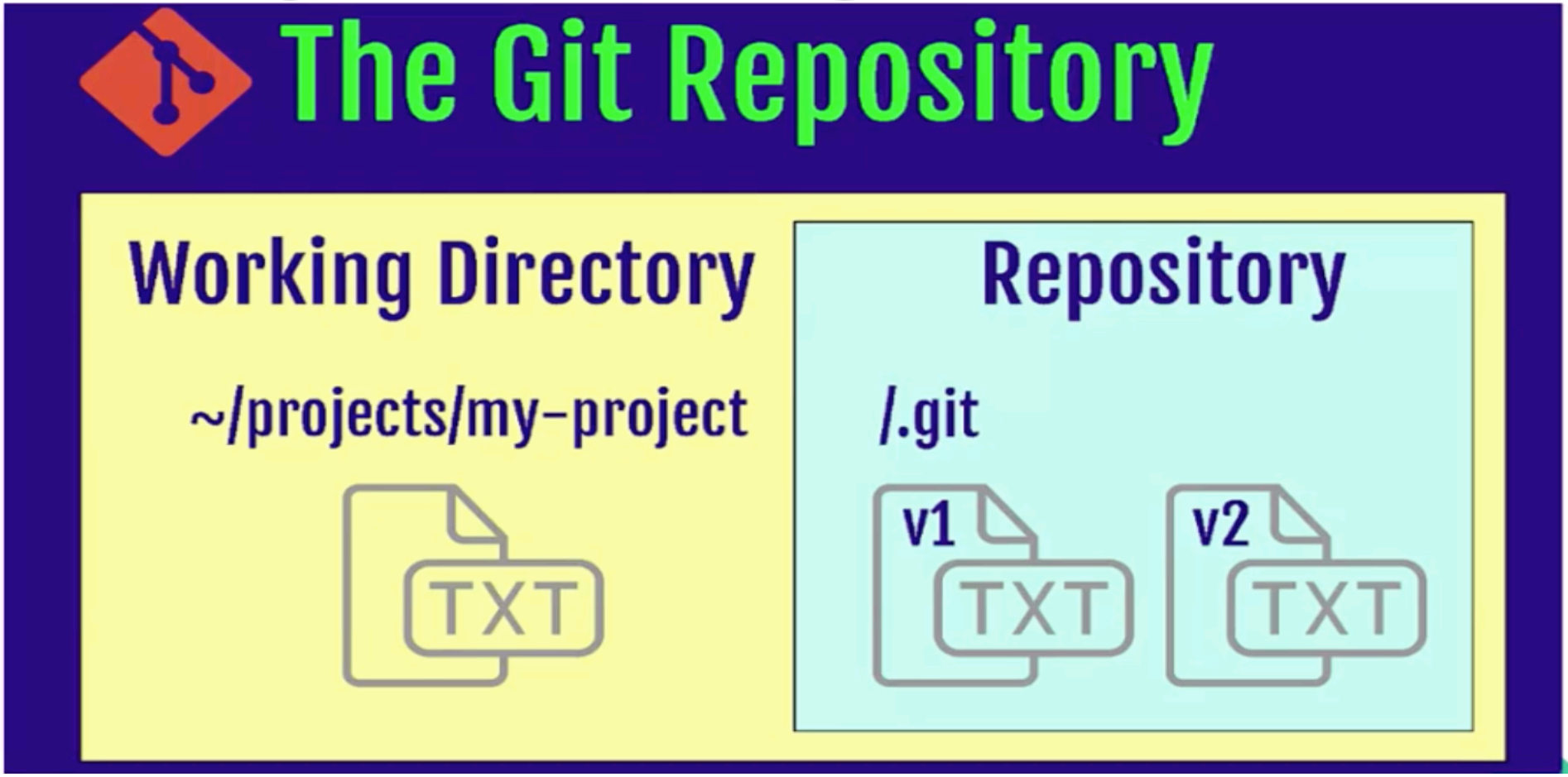
Why Git?

- **Git is a de facto standard**
  - Git is the most broadly adopted tool of its kind. This is makes Git attractive for the following reasons.
  - Vast numbers of developers already have Git experience and a significant proportion of college graduates may have experience with only Git. While some organizations may need to climb the learning curve when migrating to Git from another version control system, many of their existing and future developers do not need to be trained on Git.
  - In addition to the benefits of a large talent pool, the predominance of Git also means that many third party software tools and services are already integrated with Git including IDEs

Why Git?

- **Git is a quality open source project**
  - Git is a very well supported open source project with over a decade of solid stewardship. The project maintainers have shown balanced judgment and a mature approach to meeting the long term needs of its users with regular releases that improve usability and functionality. The quality of the open source software is easily scrutinized and countless businesses rely heavily on that quality.
  - Git enjoys great community support and a vast user base. Documentation is excellent and plentiful, including books, tutorials and dedicated web sites.
  - Being open source lowers the cost for hobbyist developers as they can use Git without paying a fee. For use in open-source projects, Git is undoubtedly the successor to the previous generations of successful open source version control systems, SVN and CVS.

Basics of GIT



Basics of GIT

There are 3 stages of GIT(local)

- **Working Directory**
  - This is the place where developers save their source code in their local system which needs to be committed later in to a repository
- **Staging Area(Pre-Commit Holding area)**
  - This is the area where you have committed your changes on your local repository but needs to be ultimately stored in a remote repository like Github
- **Commit**
  - Commits your code to the local repository
- **Remote Repository**
  - Location of the repository on a remote server (GitHub)
- **Master Branch**

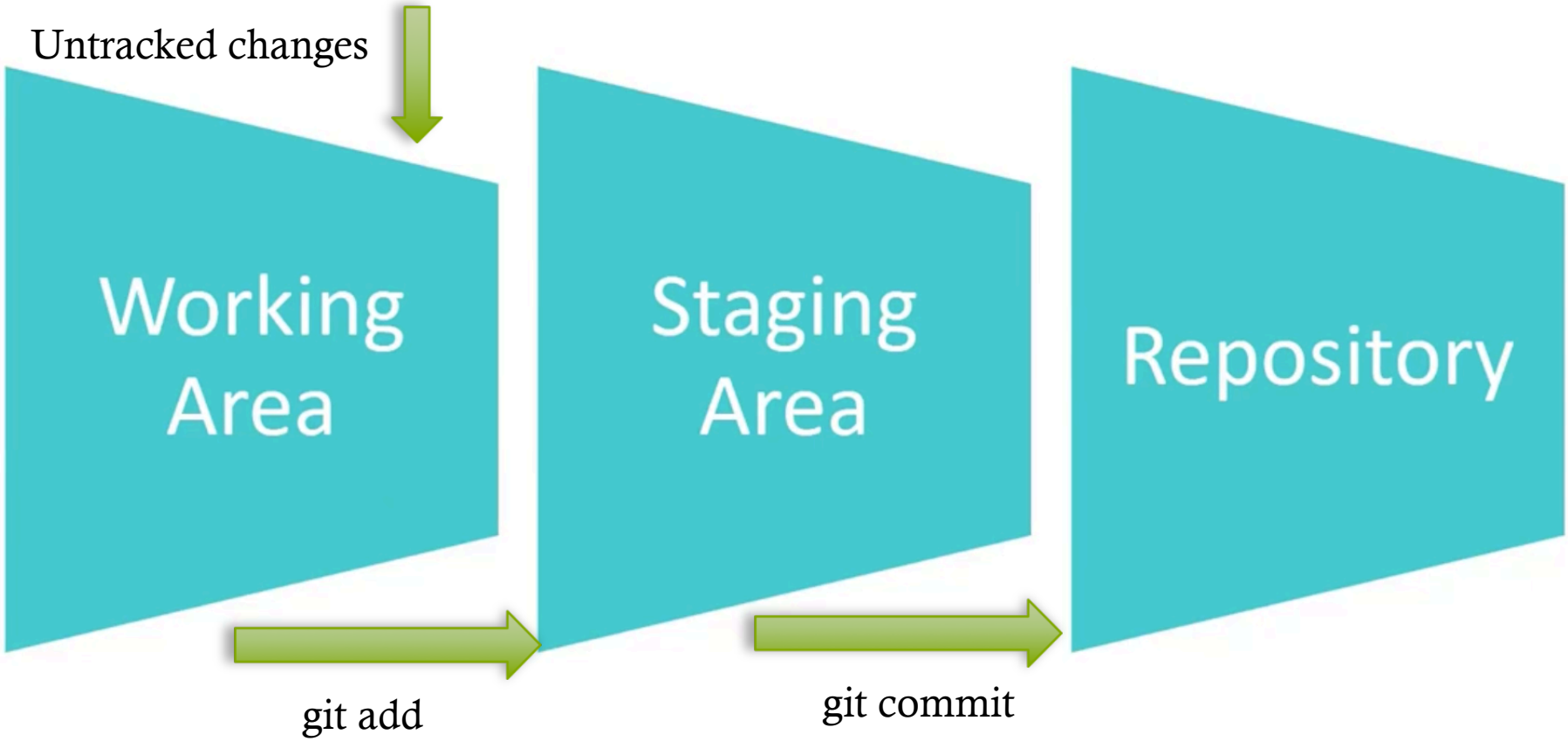
Installation of GIT

To install git in CentOS 6, execute the following command in linux terminal

$ sudo yum install git –y

$ git version

GIT Workflow



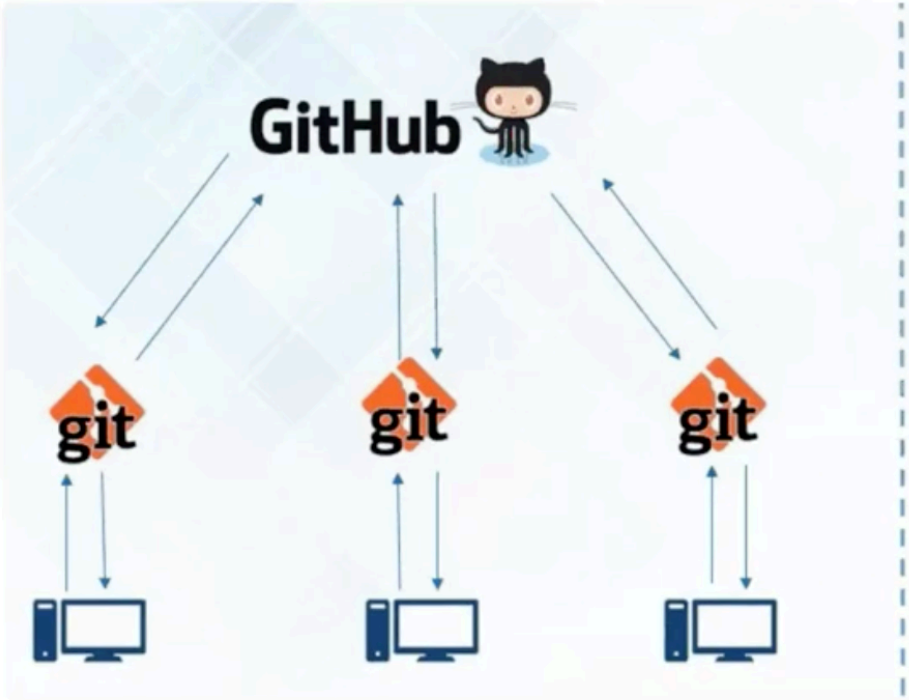
Text Editor for GIT

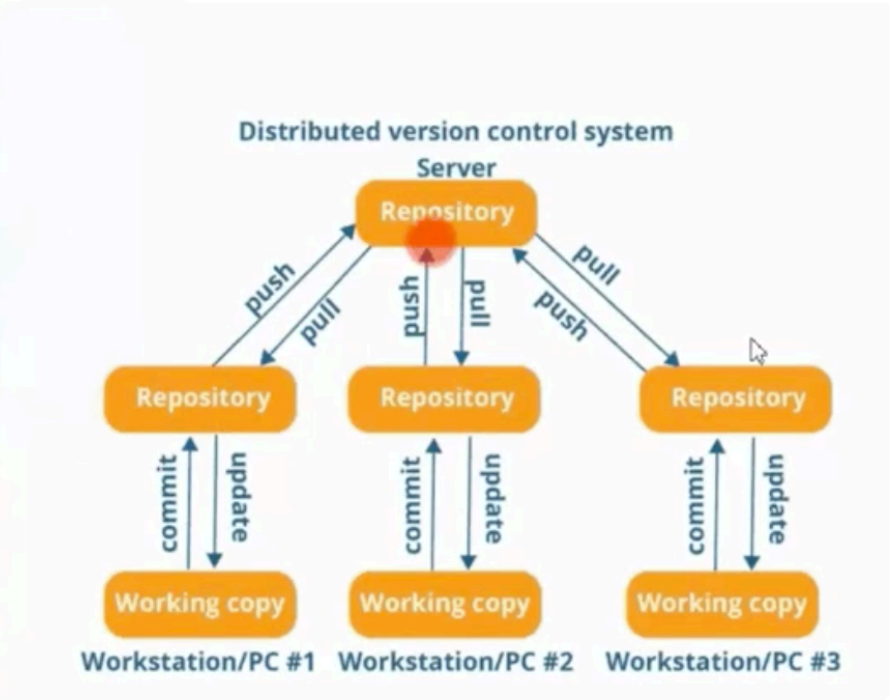
- Windows
  - Notepad++
  - Bash Alias
- MacOS
  - TextMate2
- Linux
  - Sublime Text

03.1 Exercise-Git Basic Commands

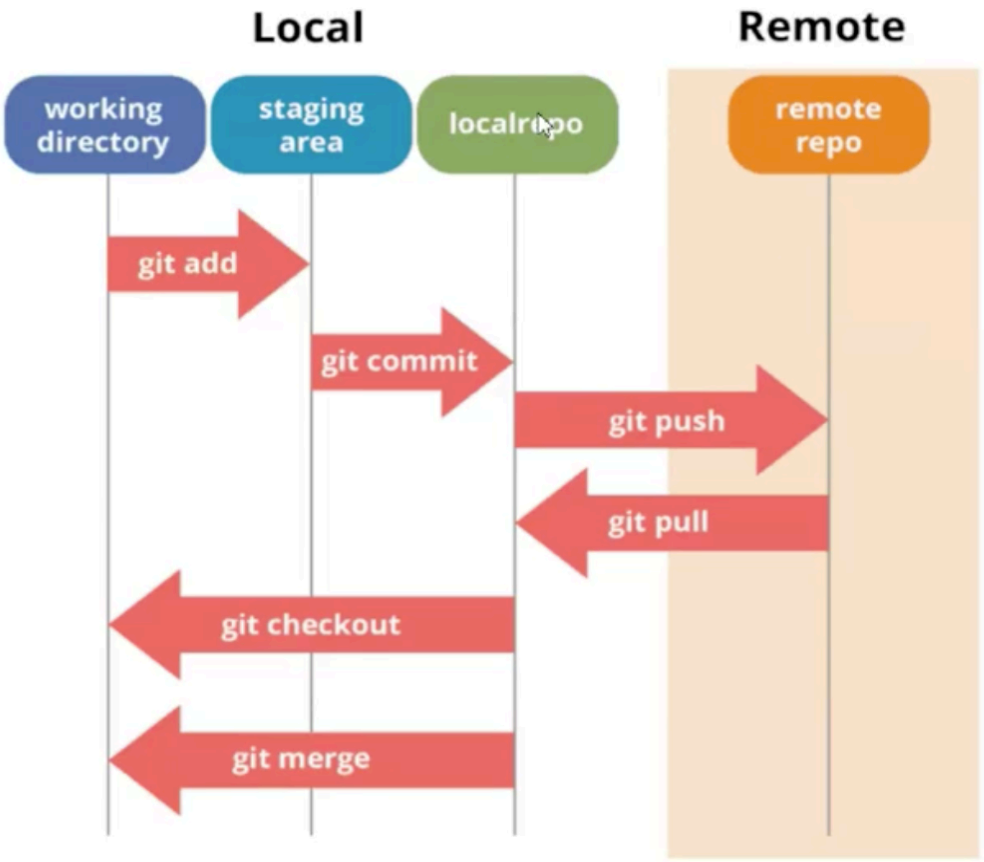
- Install Git
- Creating and initializing a local repository
- Create files and add to staging of Git
- Commit to local repository
- Viewing the commit logs
- Git diff command
- Git delete command

GIT Vs GitHub





GIT Workflow



Workflow

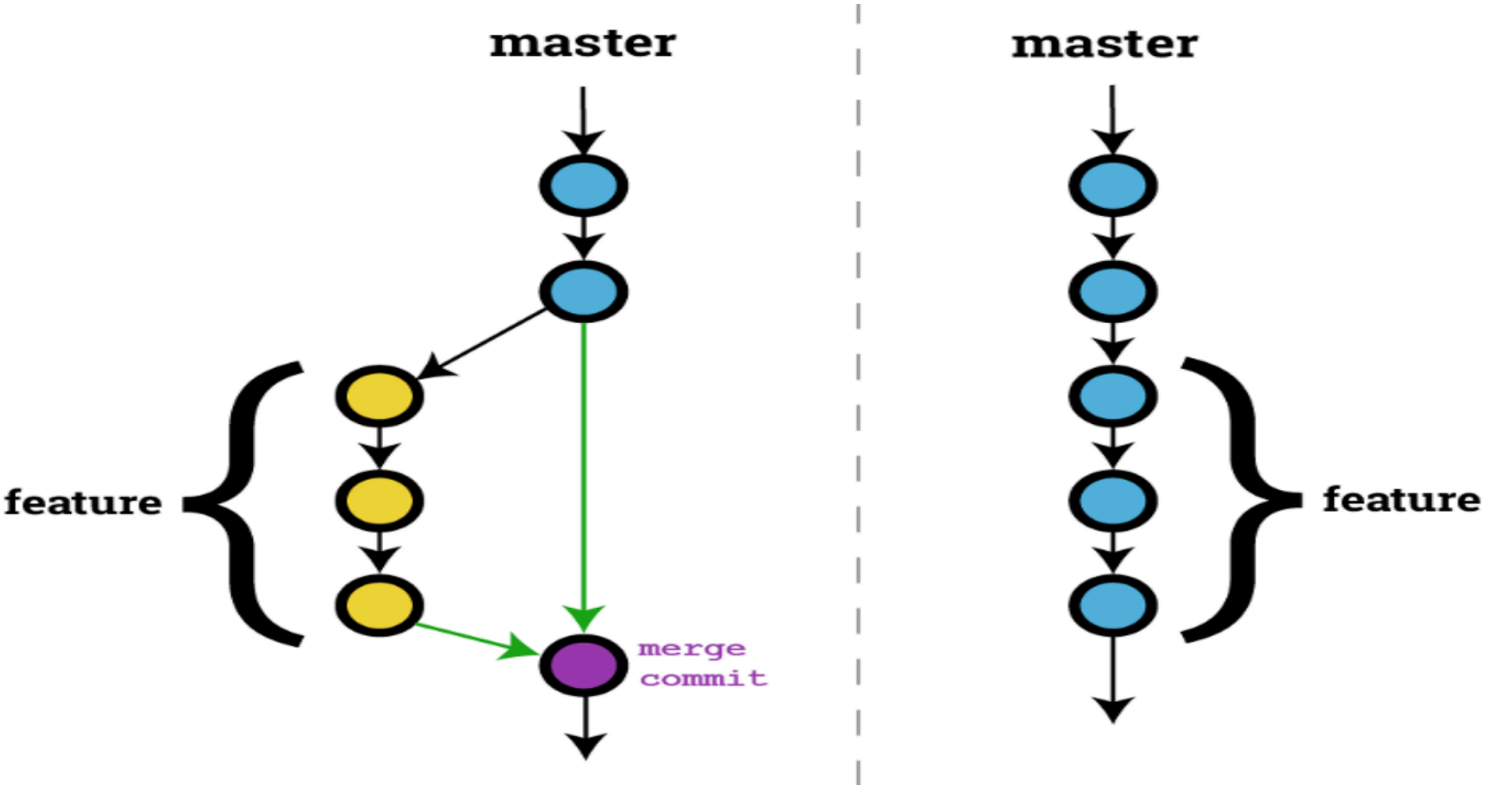
GIT & GitHub

- GitHub is a repository hosting service. Think of it as the “cloud for the code”
- Git Hub is the Central Repository
- Git is client installed on each of the developer machines.
- Git synchronizes your local repository and central repository.

03.2 Exercise-Creating Git Central Repository

- Creating a Central Repository in GitHub
- Synchronizing the central repo with local repo
- Pushing new changes in local repo to central repo

GIT Branches



GIT Branches

- GitHub is a repository hosting service. Think of it as the “cloud for the code”
- Git Hub is the Central Repository
- Git is client installed on each of the developer machines.
- Git synchronizes your local repository and central repository.

AWS CodeCommit

- Git repositories can be expensive.
- The industry includes:
  - GitHub: free public repositories, paid private ones
  - BitBucket
  - Etc...

AWS CodeCommit

- Private Git repositories
- No size limit on repositories (scale seamlessly)
- Fully managed, highly available
- Code only in AWS Cloud account => increased security and compliance
- Secure (encrypted, access control, etc...)
- Integrated with Jenkins / CodeBuild / other CI tools

03.3 Exercise - CodeCommit

- Creating a AWS CodeCommit repository in AWS
- Accessing the repository from an ec2 instance
- Updating the repository with custom code
- Creating new branches and Merging
- Creating triggers and Notifications

03.4 Exercise – CodeCommit-Lambda

- Integrate CodeCommit with Lambda