Git and GitHub Training

©Simplilearn. All rights reserved.

simplilearn

Git Basics

©Simplilearn. All rights reserved.

Learning Objectives

By the end of the lesson, you will be able to:

- Explain Git and version control system
- Define Git buzzwords
- Migrate from SVN and Perforce
- Illustrate Git configuration level and basic commands
- Define web-scale architecture
- Differentiate GitHub, GitLab, and Bitbucket

©Simplilearn. All rights reserved.

Overview of Version Control Systems

©Simplilearn. All rights reserved.

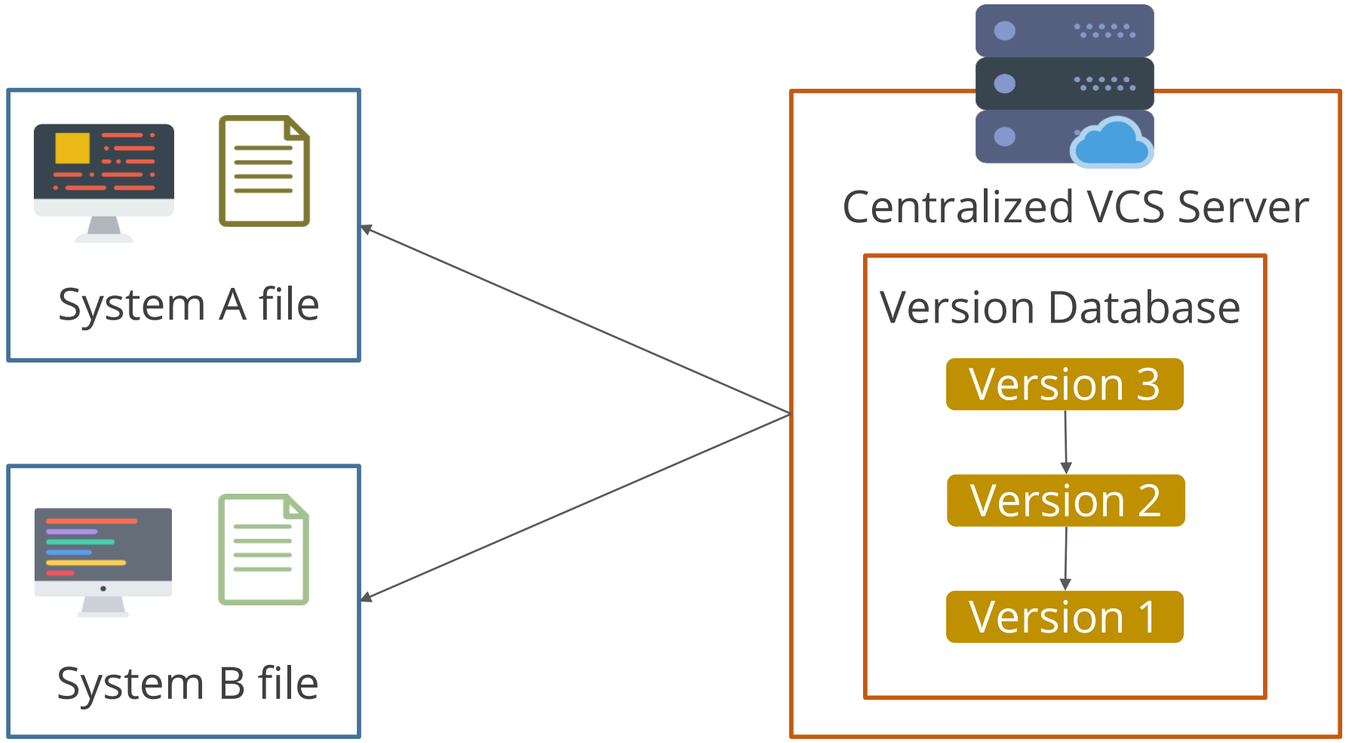
Version Control Systems: Definition

- Version control is a system that records changes to a set of files over a period of time to recall specific versions.
- Version Control System (VCS) can be used to store every version of an image or layout.

For example:

©Simplilearn. All rights reserved.

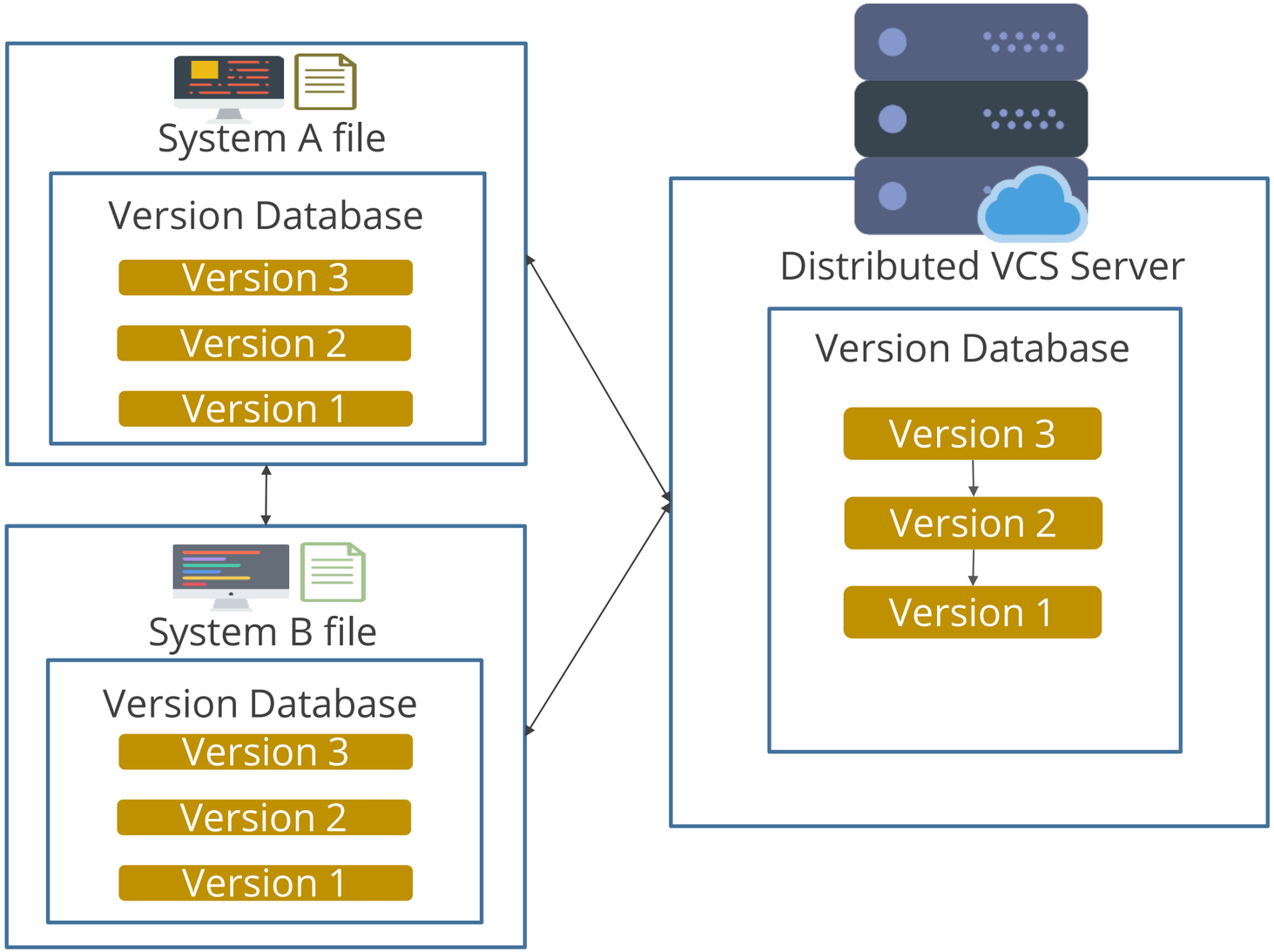
Centralized Version Control System



- Uses a central server to store all the files.
- Performs every operation directly on the repository.
- Stores file versions on the central VCS server.
- For example: Tortoise SVN

©Simplilearn. All rights reserved.

Distributed Version Control System



- Moves from the client-server approach to peer-to-peer approach.
- Updates the local repositories with new data from the central server. The changes get reflected to the main repository.
- For example: Git

©Simplilearn. All rights reserved.

Version Control System: Benefits

Collaboration:
The team can work on any file at any instance and later allows to merge all the changes into a common version.

Storage:
Acknowledges that there is only one project whereas all the past versions and variants are neatly packed up inside the VCS

Backup:
Distributed VCS like Git act as a backup

©Simplilearn. All rights reserved.

Version Control System: Tools

Some of the preferred open-source version control system tools for easier set up are:

©Simplilearn. All rights reserved.

History of Git

©Simplilearn. All rights reserved.

Git: History

Linus Torvalds
(Creator of Linux)

Initially, it was developed to manage the Linux development community

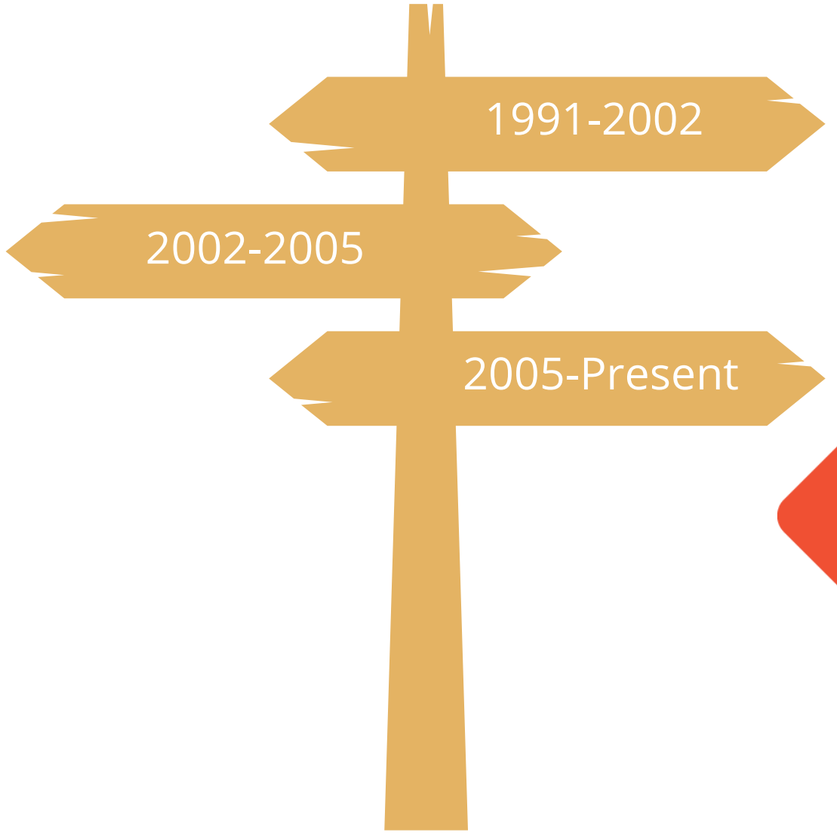
Originally, it was written in C language and then re-implemented in other languages

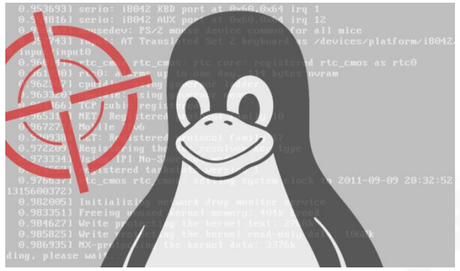
©Simplilearn. All rights reserved.

Git: History

Linux code has been managed using:

BitKeeper



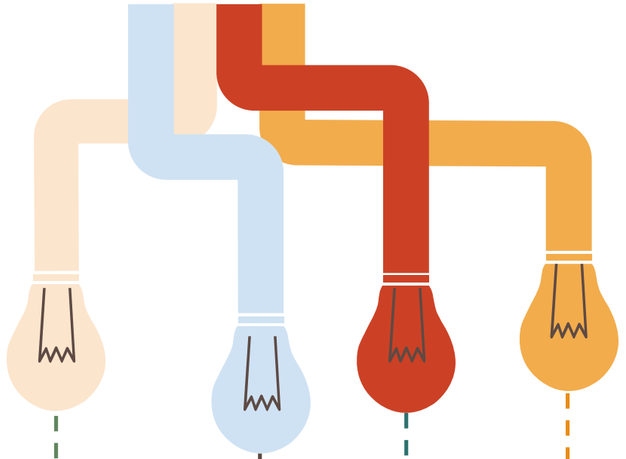


An archive of patches

Git

©Simplilearn. All rights reserved.

Importance of Git



Has flexile environment

Keeps the files in a secure server

Creates and manages remote repositories

Facilitates branching and merging

©Simplilearn. All rights reserved.

simplilearn

Overview of Git

©Simplilearn. All rights reserved.

Limitations of Existing Version Control Systems

- Single server
- Centralized architecture
- Offline changes are not possible

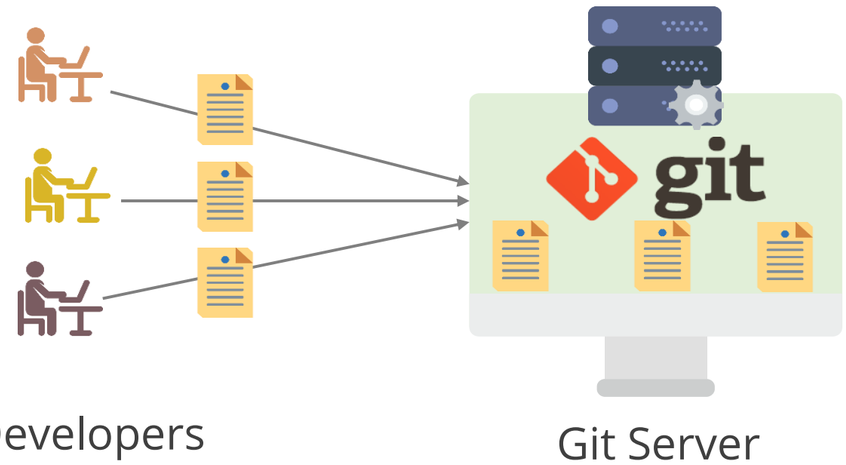
Retrieve files → Make changes → Commit

- All changes made in central server
- Error handling

©Simplilearn. All rights reserved.

Git: Definition

Git is a Version Control System for tracking changes in computer files. It is generally used for source code management in software development.



- Tracks changes in the source code
- Uses distributed version control tool for source code management
- Allows multiple developers to work together
- Supports non-linear development because of its several parallel branches

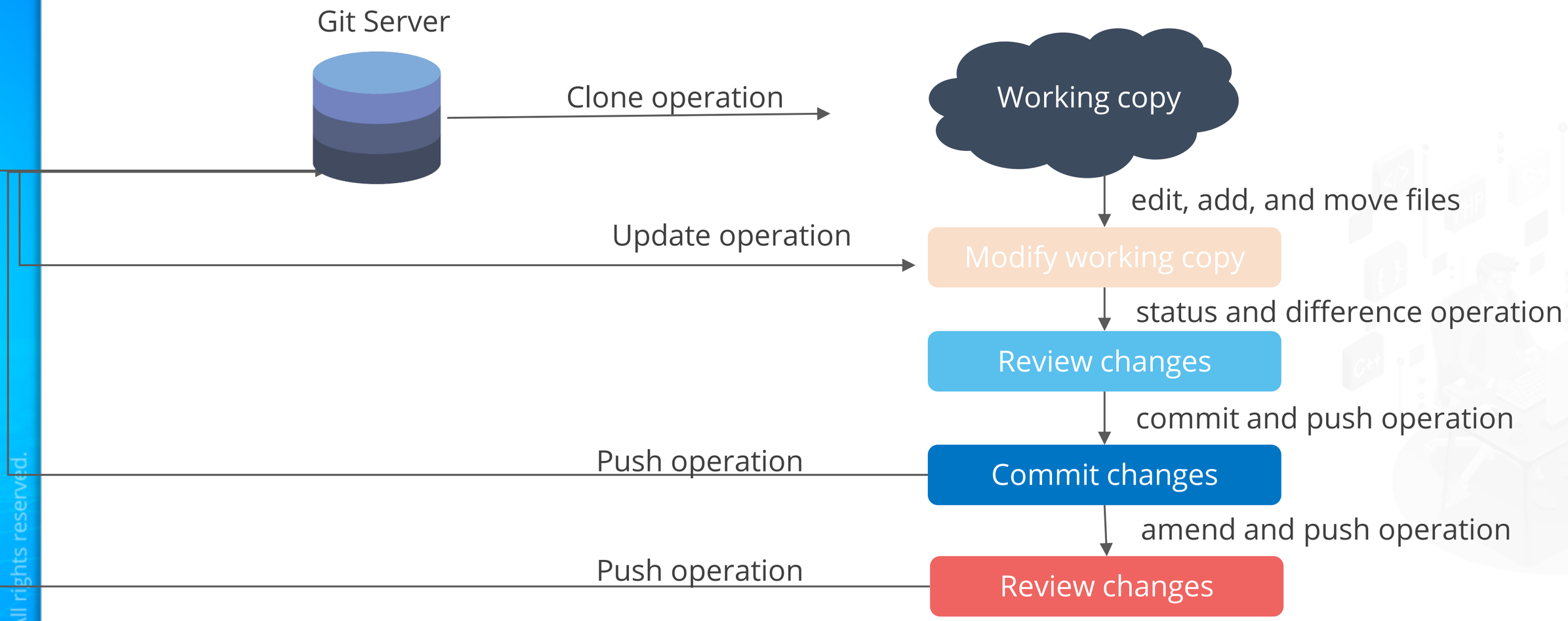
©Simplilearn. All rights reserved.

Git: Features

1. Compatible
2. Distributed
3. Nonlinear
4. Branched
5. Lightweight
6. Fast
7. Open Source
8. Reliable
9. Secure
10. Economical

©Simplilearn. All rights reserved.

Git: Life Cycle



©Simplilearn. All rights reserved.

Git vs. Other Version Control Systems

Git	Other Version Controls Systems
The users or the team maintains its own repository, instead of working from a central repository.	The users or the team uses a central code repository model.
The changes are stored as patches or can be characterized as sets.	The storage of changes in the central repository depends on the user's interest.
They focus on patches or change sets as a discrete unit that can be exchanged between repositories.	They track changes from version-to-version of different files or states of the directory.
They do not require central server.	They need a central server.
There is a no single point of failure.	There is a single point of failure.

©Simplilearn. All rights reserved.

Git vs. GitHub

Git	GitHub
It is installed and maintained on the local system.	It is hosted on the web.
It is a command line tool.	It is a graphical interface.
It is a tool to manage different versions of the file in a git repository.	It is a space to upload a copy of the git repository.

©Simplilearn. All rights reserved.

Install Git on Linux

Problem Statement: Installation of Git on Linux platform.

Steps to Perform:

1. Install Git
2. Verify the installation

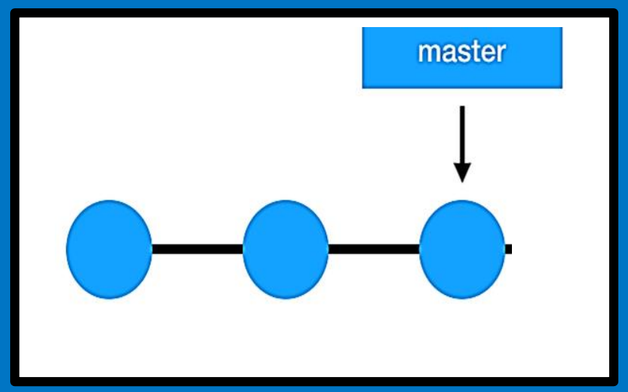
UNASSISTED PRACTICE

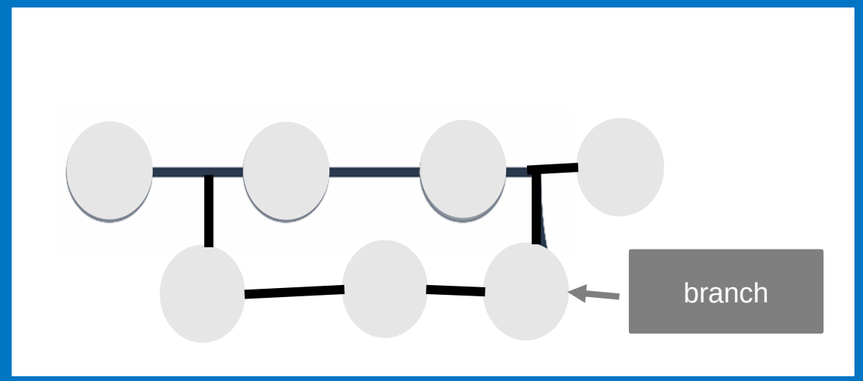
©Simplilearn. All rights reserved.

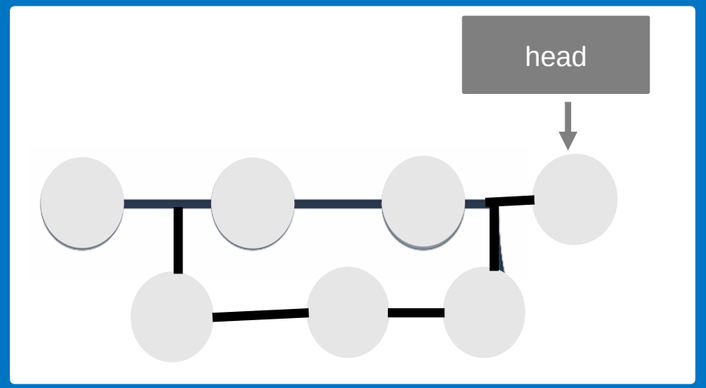
Overview of Git Buzzwords

©Simplilearn. All rights reserved.

Git Buzzwords



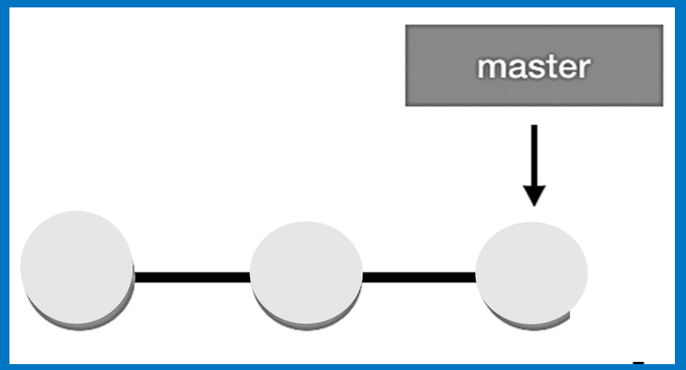


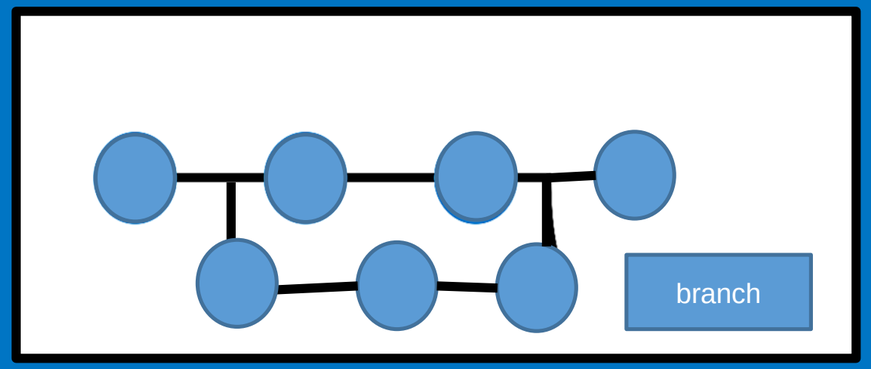


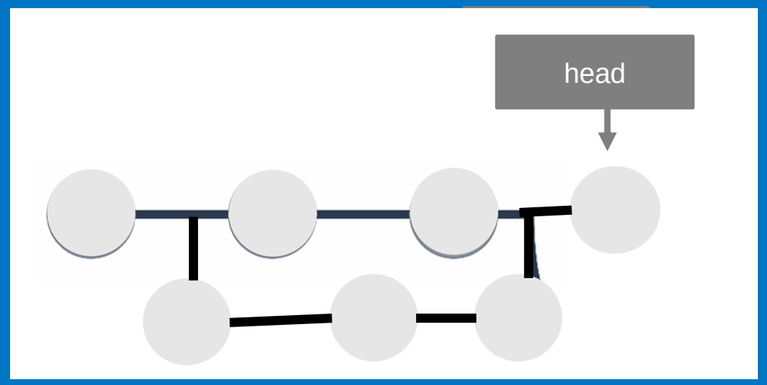
- It is a default branch.
- It is used by CI tools for build and deployment.
- It is followed by the other repositories.

©Simplilearn. All rights reserved.

Git Buzzwords



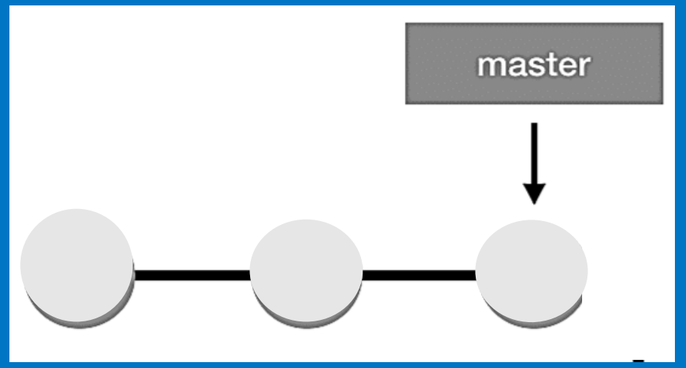


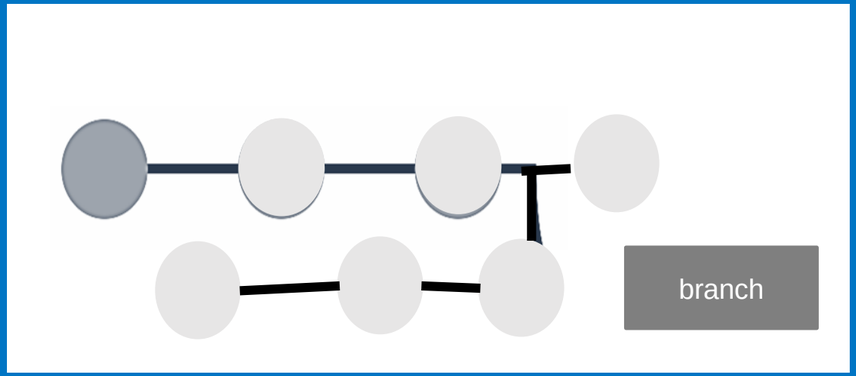


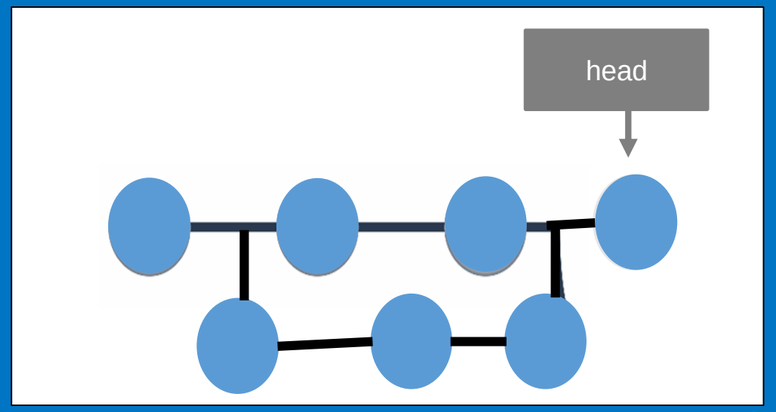
- It is a light weight working copy.
- It has a staging area.
- It works without impacting the master branch.

©Simplilearn. All rights reserved.

Git Buzzwords



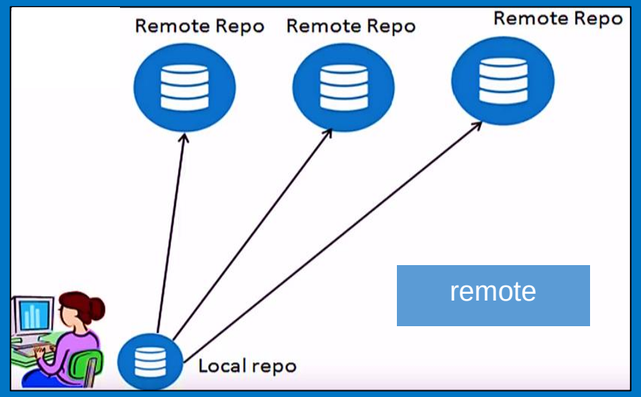


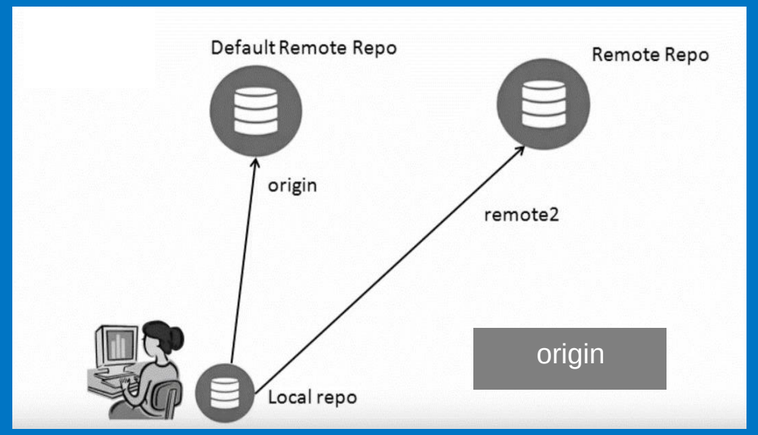


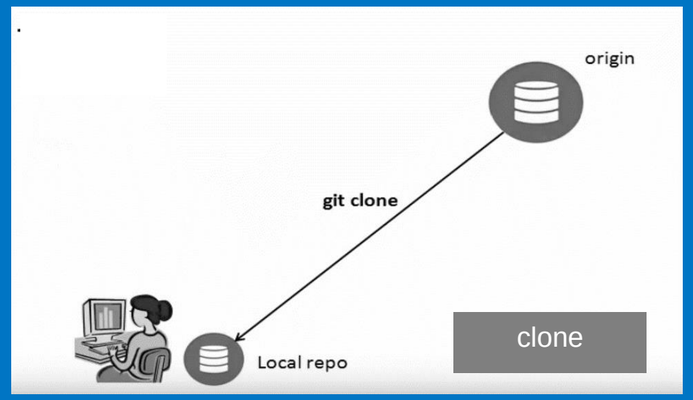
- It is a pointer to the latest commit of the working branch.
- It is present on every repository.
- It will point to the latest commit during branch switch.

©Simplilearn. All rights reserved.

Git Buzzwords



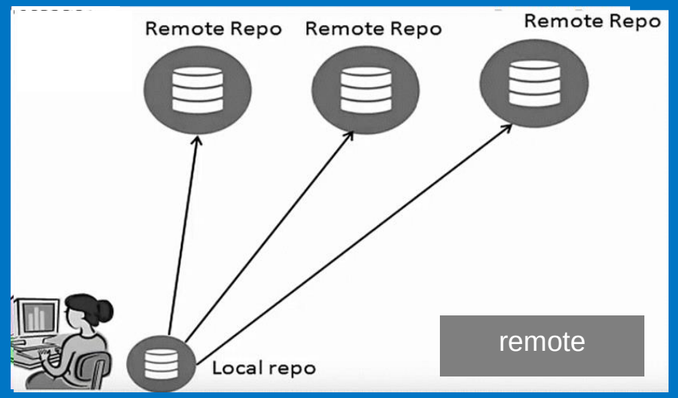


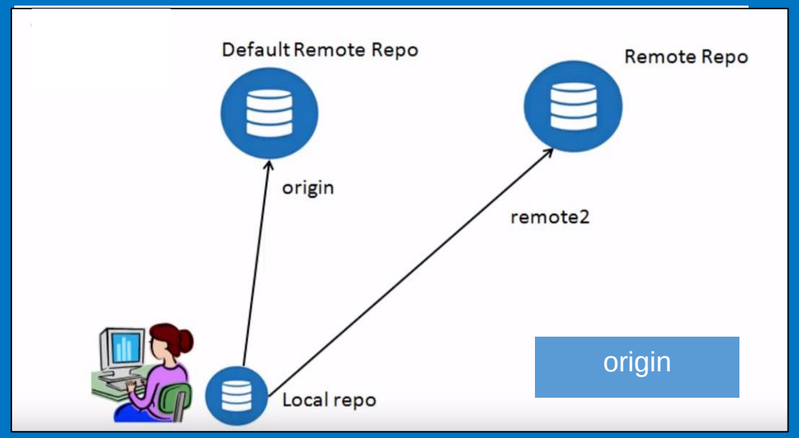


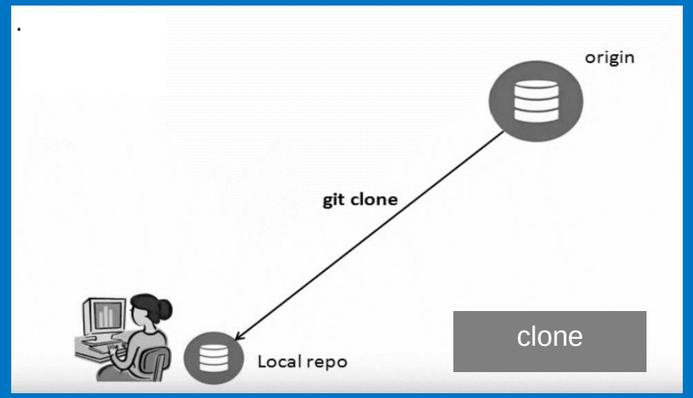
- It is a git repository on a network outside the local machine.
- It can have more than one remote repositories pointing from the local repository.
- It can be managed and referenced with short names.

©Simplilearn. All rights reserved.

Git Buzzwords



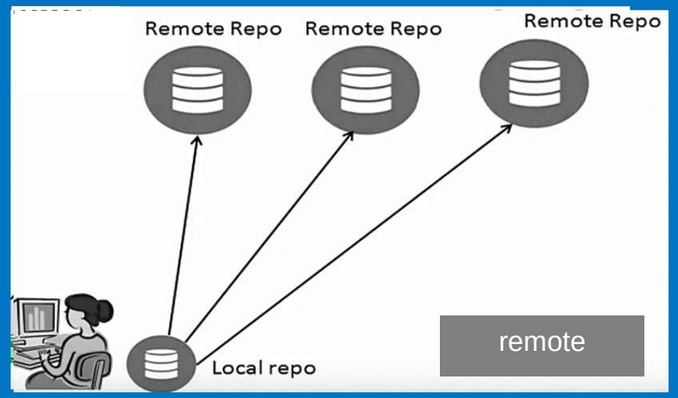


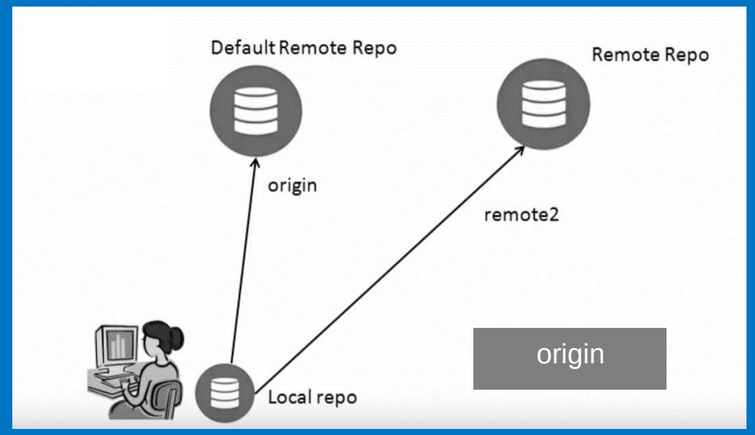


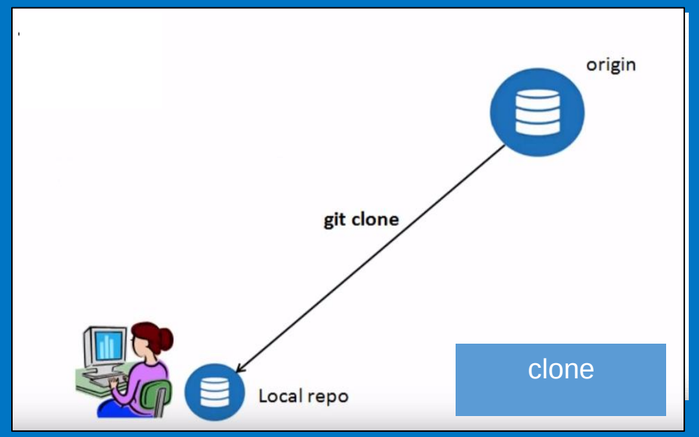
- It associates remote repository with names.
- It is a local name set for the default repository.
- It is useful to point the default repository when executing git commands.

©Simplilearn. All rights reserved.

Git Buzzwords



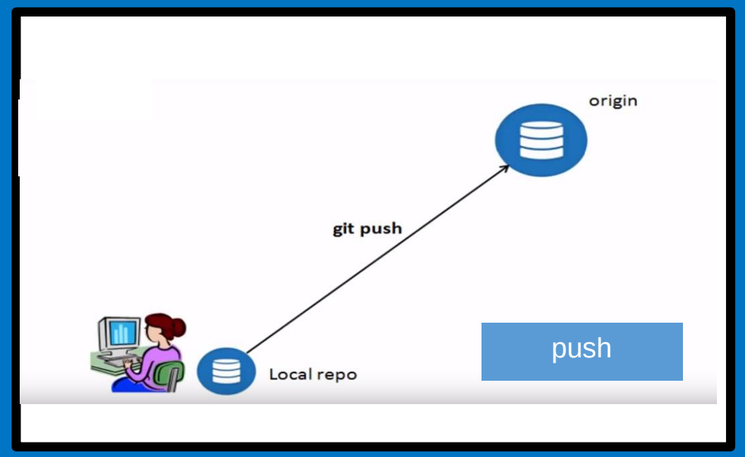


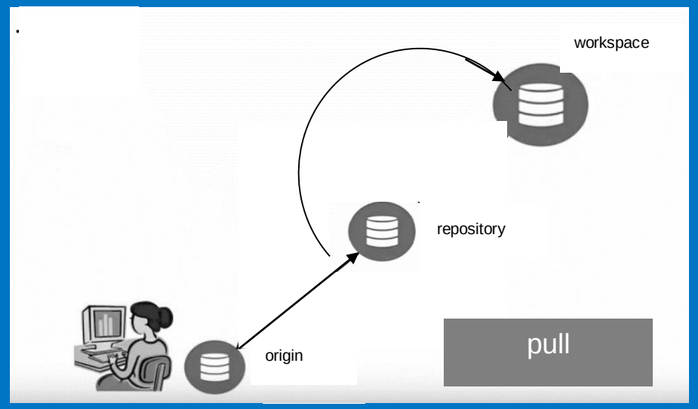


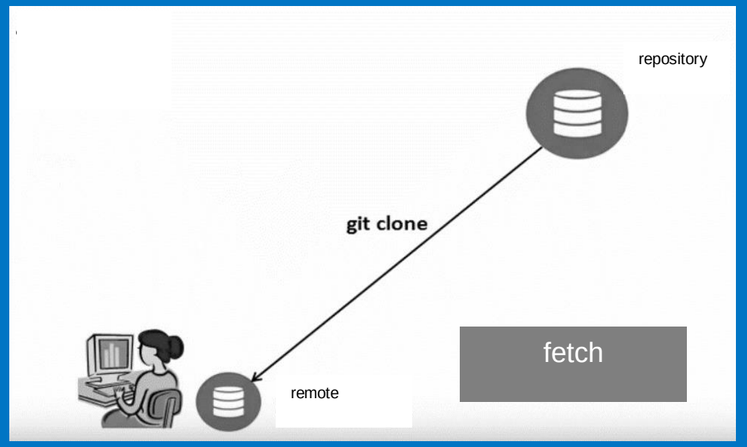
- It copies the existing repository from a remote repository.
- It will get the complete repo, whereas checkout will only fetch the working copy.
- It helps to replicate the repo on the local machine.

©Simplilearn. All rights reserved.

Git Buzzwords



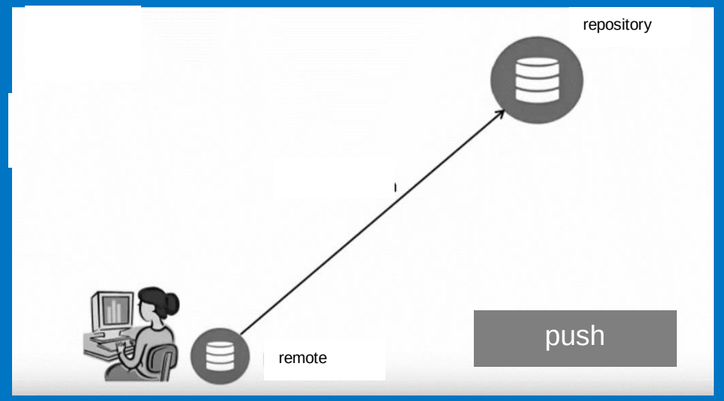


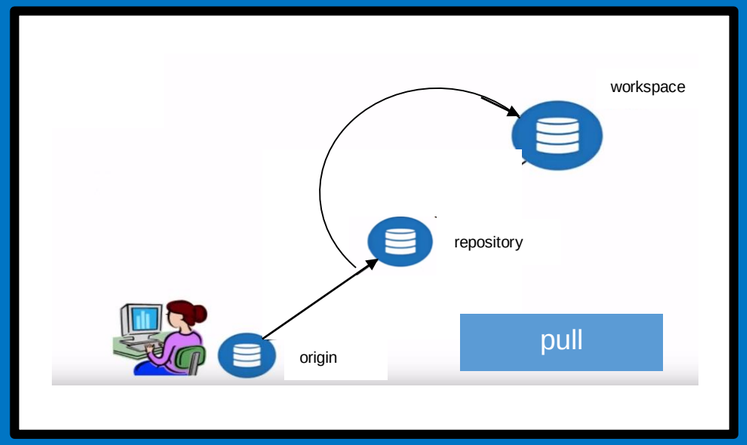


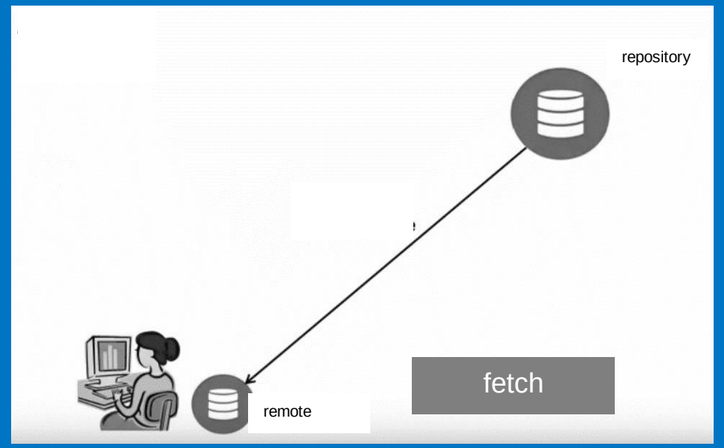
- It pushes changes from the local to the remote repository.
- It is performed after committing the changes to the local repository.
- It syncs the changes with the local and remote repository.

©Simplilearn. All rights reserved.

Git Buzzwords



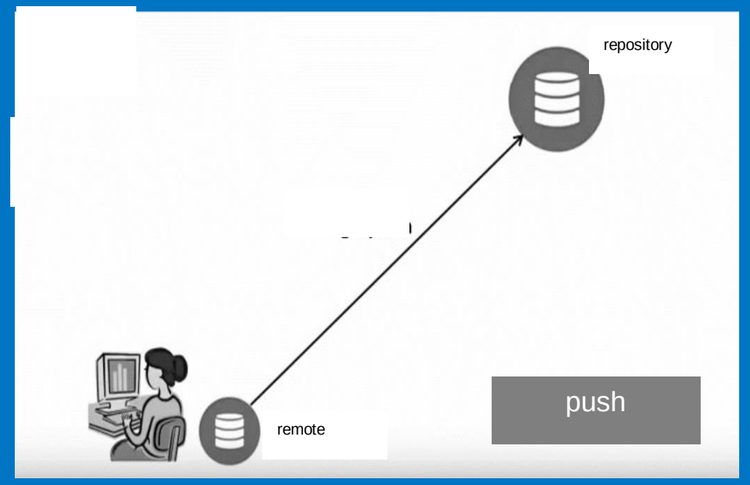


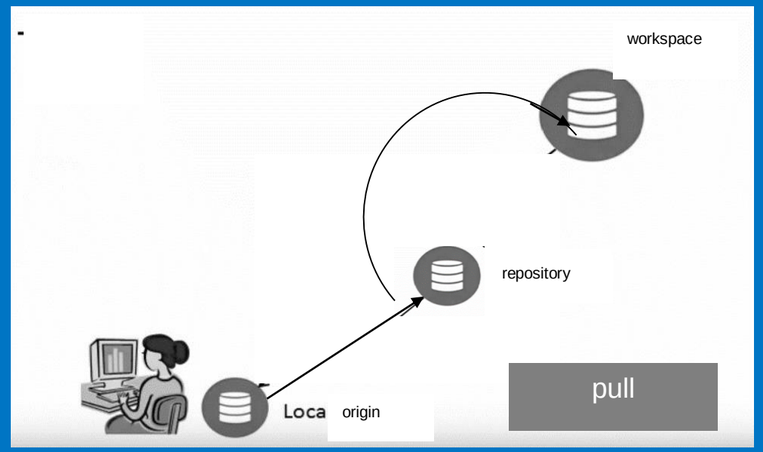


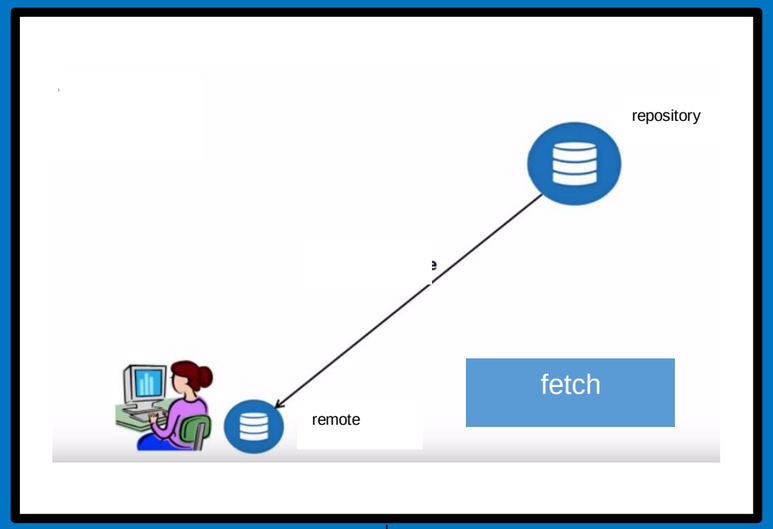
- It transfers the updates from the local to the remote repository.
- It syncs the changes from remote to the local repository.
- It takes current code from remote repository and merges the change with the local repository.

©Simplilearn. All rights reserved.

Git Buzzwords



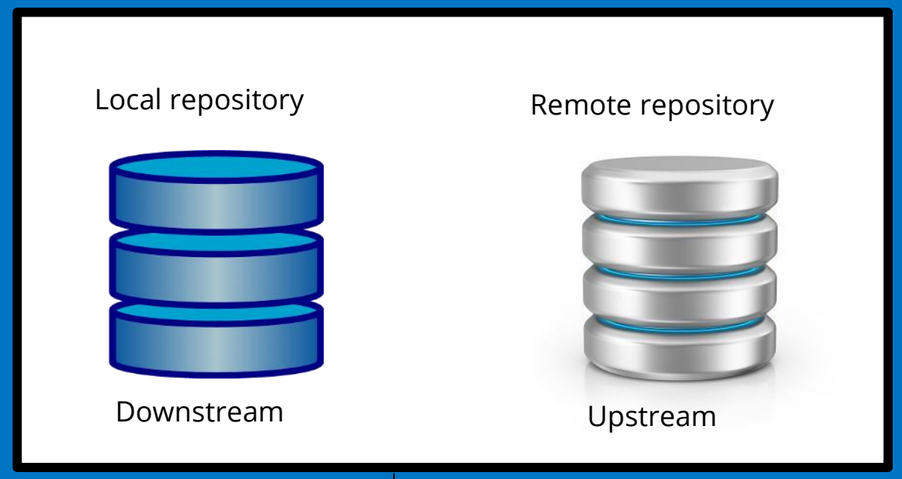




- It will not merge the changes with a local repository.
- It gives updates from remote to the local repository.
- It syncs the changes from remote to the local repository.

©Simplilearn. All rights reserved.

Git Buzzwords



- When the data is flowing between repository A and B, repository A is upstream and B is downstream making B pull data from repository A.

©Simplilearn. All rights reserved.

Migration from SVN to Git

©Simplilearn. All rights reserved.

SVN: Definition

- It is a centralized version control system
- It is an open source characterized by its reliability
- It supports the needs of a wide variety of users and projects
- It has all the source files and versions of the files

Drawback

It has a tedious branching model which adds complexity implementing a branch strategy

©Simplilearn. All rights reserved.

Git vs. SVN

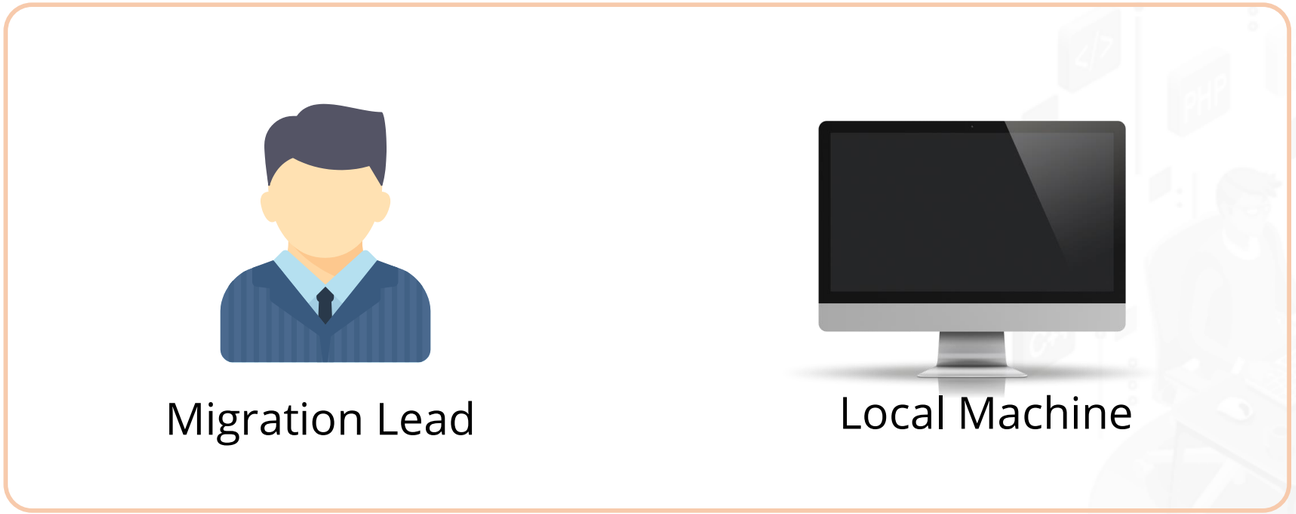
Git	SVN
Used by 70% of the developers	Used by 25% of the developers
Supports distributed version control	Supports centralized version control
Can work locally (offline)	Must be connected to commit
Each user has a copy of the full repository	Each user has a copy of only the trunk
Easy to fork, branch, and merge	Easy to branch and merge

©Simplilearn. All rights reserved.

Migrate to Git from SVN

The process involves five steps:

- Prepare the environment
- Convert SVN to a local Git repository
- Synchronize the local repository
- Share the Git repository
- Migrate the development

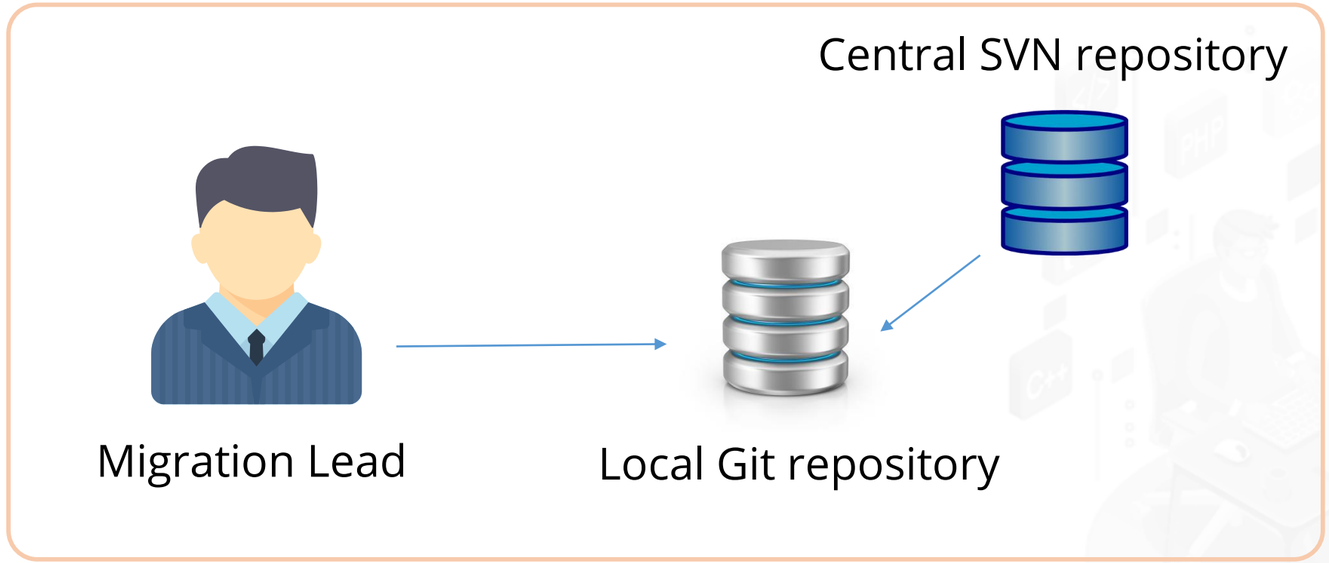


©Simplilearn. All rights reserved.

Migrate to Git from SVN

The process involves five steps:

- Prepare the environment
- **Convert SVN to a local Git repository**
- Synchronize the local repository
- Share the Git repository
- Migrate the development

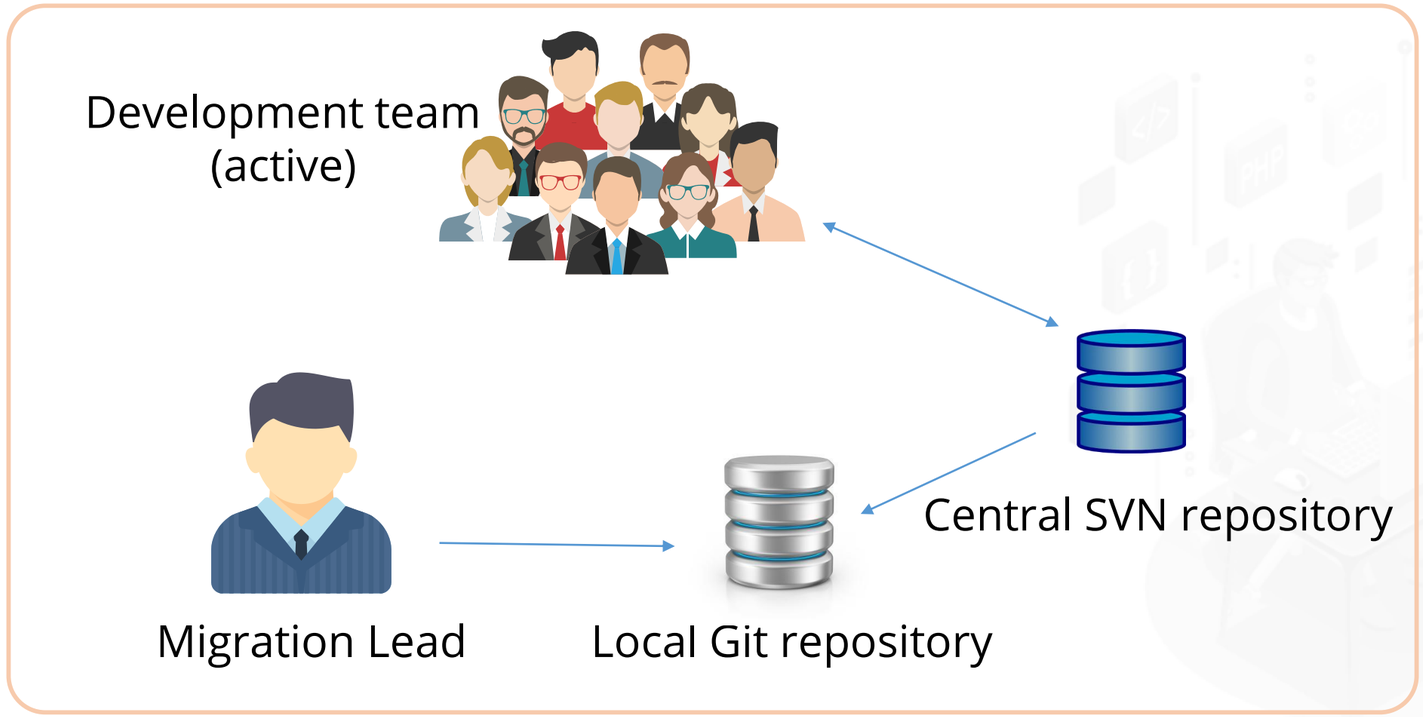


©Simplilearn. All rights reserved.

Migrate to Git from SVN

The process involves five steps:

- Prepare the environment
- Convert SVN to a local Git repository
- **Synchronize the local repository**
- Share the Git repository
- Migrate the development

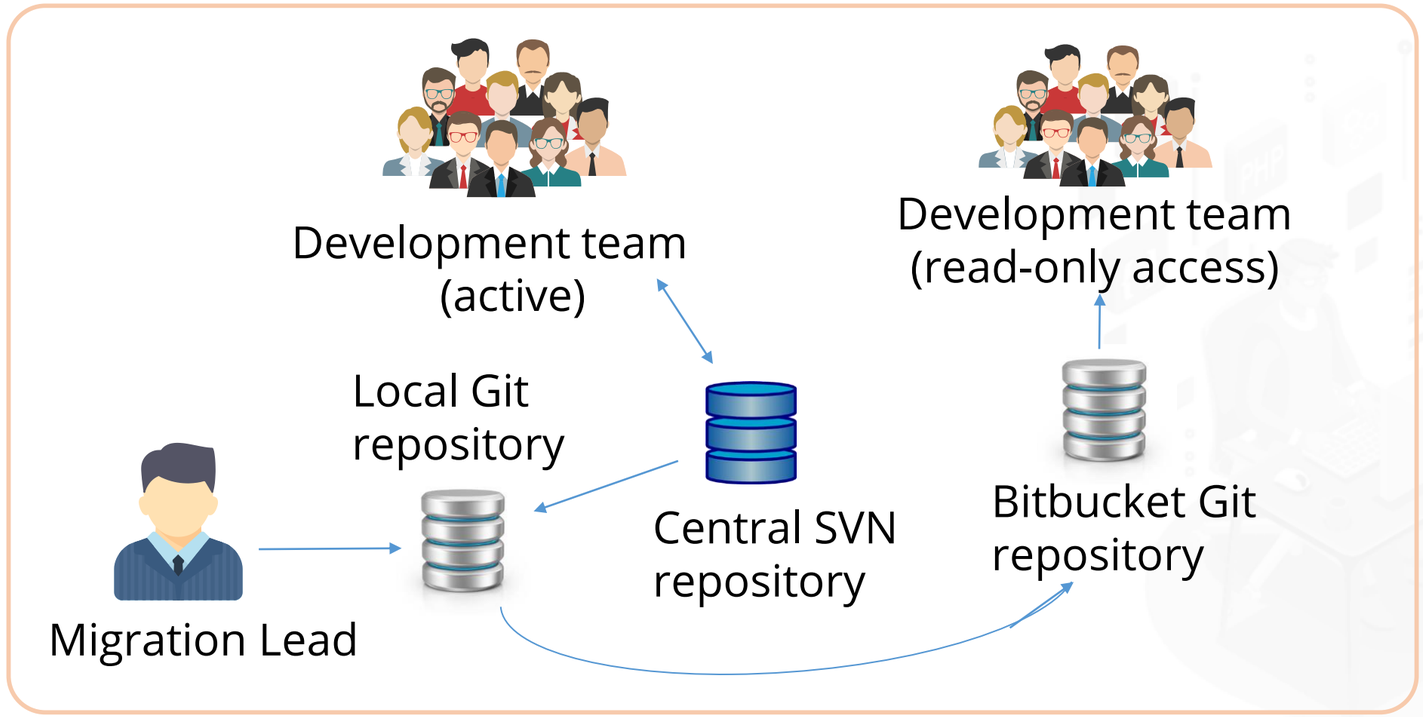


©Simplilearn. All rights reserved.

Migrate to Git from SVN

The process involves five steps:

- Prepare the environment
- Convert SVN to a local Git repository
- Synchronize the local repository
- **Share the Git repository**
- Migrate the development

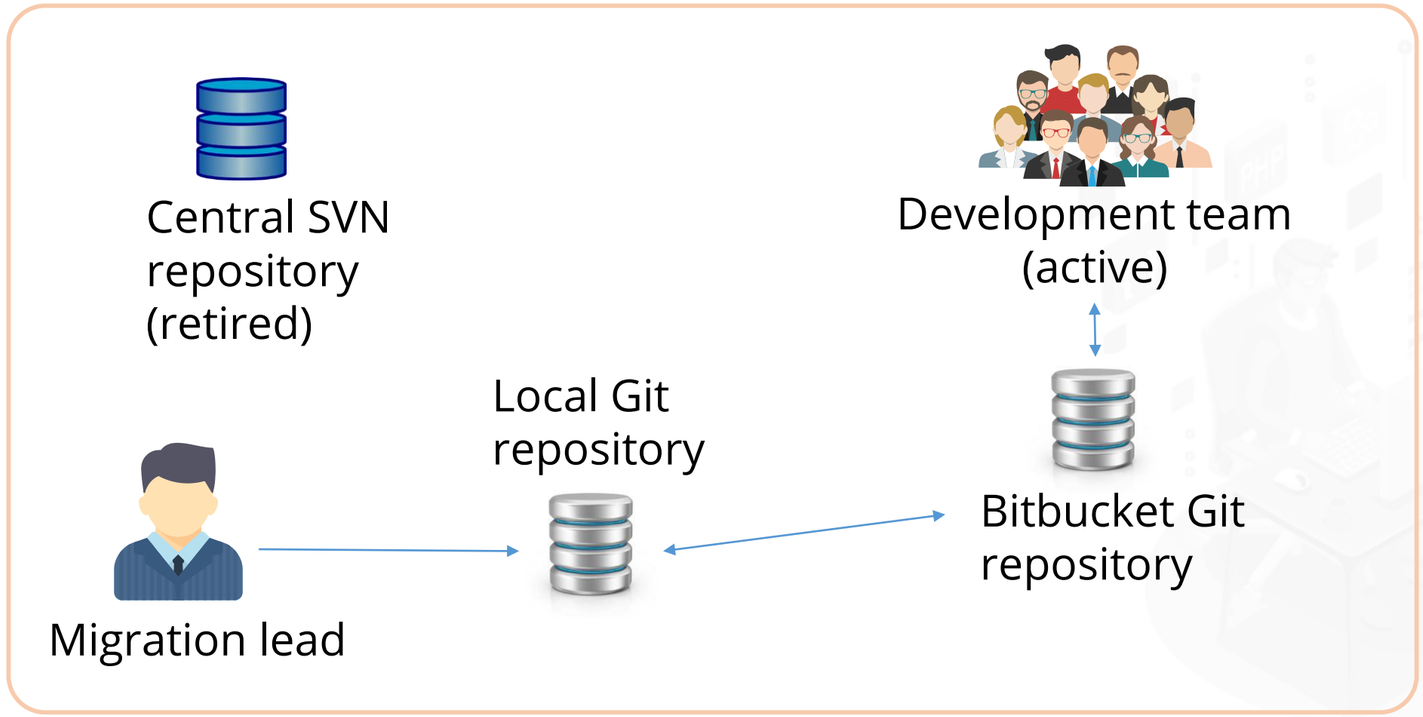


©Simplilearn. All rights reserved.

Migrate to Git from SVN

The process involves five steps:

- Prepare the environment
- Convert SVN to a local Git repository
- Synchronize the local repository
- Share the Git repository
- **Migrate the development**



©Simplilearn. All rights reserved.

Migration to Git from SVN

Problem Statement: Your team wants the file to be available offline and also have full access. So, you have to migrate to Git from SVN for sharing the project files with your coworkers.

Steps to Perform:

1. Create a repository
2. Open Git Bash
3. Clone SVN to Git
4. Add the GitHub repository
5. Push the code to remote GitHub

UNASSISTED PRACTICE

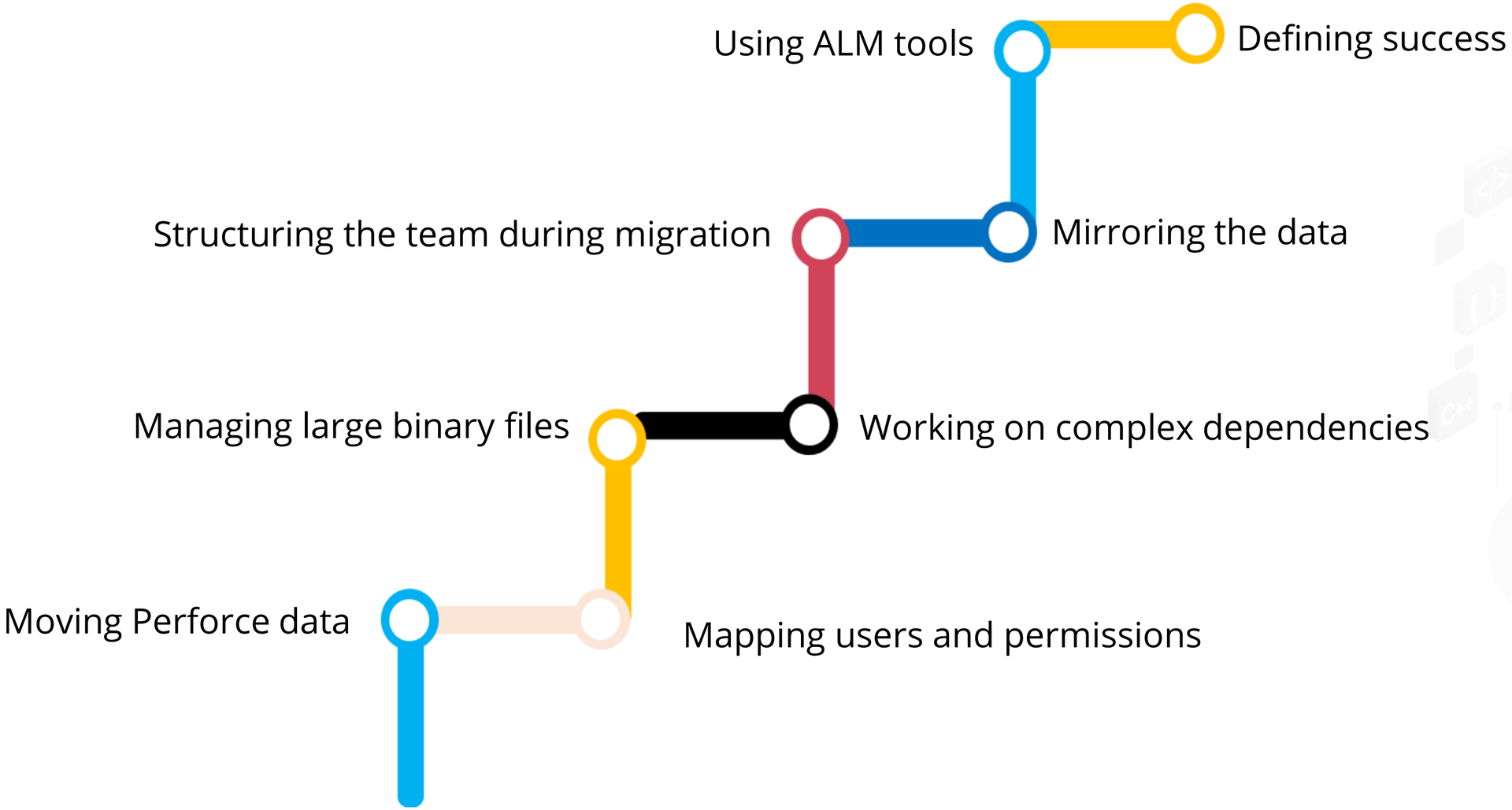
©Simplilearn. All rights reserved.

Migration to Git from Perforce

©Simplilearn. All rights reserved.

Migration to Git from Perforce

The process involves eight steps:



©Simplilearn. All rights reserved.

Perforce: Definition

- It is a commercial centralized version control system
- It has the flexibility of collaborating on the same codebase and code reviews
- It can work on large projects
- It maintains a branching record on a per-file basis

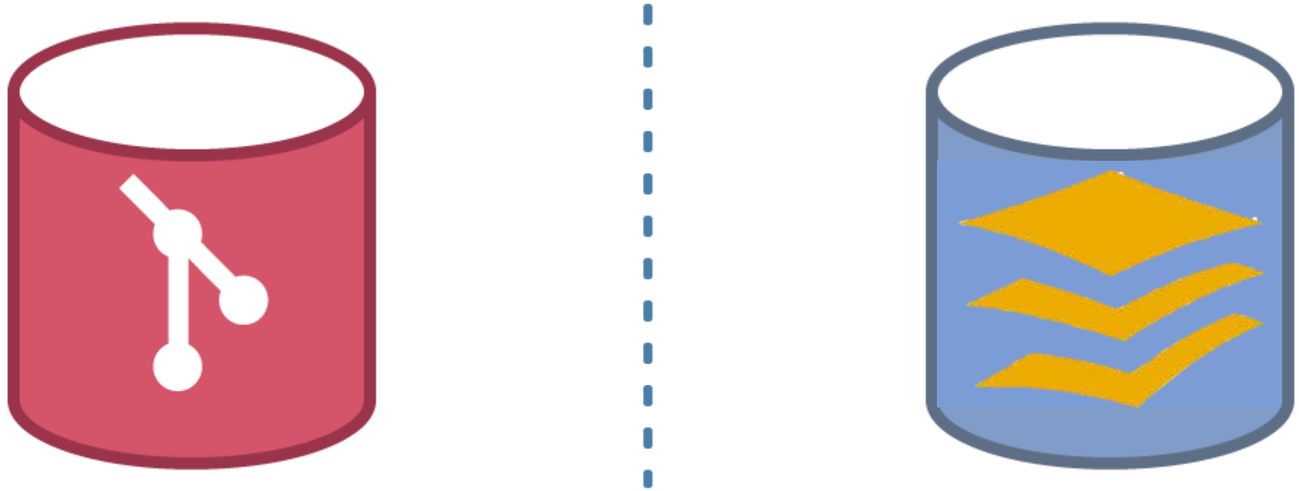
Drawback

It is not free of cost

©Simplilearn. All rights reserved.

Git vs. Perforce

Git	Perforce
Supports distributed version control	Supports commercial centralized version control
Offers large file system	Offers all files in one repository
It is free of cost	It is not free of cost
Coordinates merge conflicts across repository	Scalable with cross repositories



©Simplilearn. All rights reserved.

Migration to Git from Perforce

The process involves eight steps:

- Moving Perforce data
- Mapping users and permissions
- Managing large binary files
- Working on complex dependencies
- Structuring the team during migration

There are two general approaches for moving the data over from Perforce to Git:

- **Git Fusion:** If you want to preserve the entire history of the data, Git Fusion tool can be used to extract a section of a Perforce server into a Git repository.
- **Start over**: If you want the simplest and fastest technique without any accumulated baggage, start over technique can be used.

©Simplilearn. All rights reserved.

Moving Perforce Data: Git Fusion

The steps involved in the Git fusion process are:

Step 01
Install Git Fusion

Step 02
Set up the correct view of data

Step 03
Use Git client to clone from Git Fusion

Step 04
Push the repository into Bitbucket

- It requires the maximum setup and runtime
- It preserves most of the history
- It maintains legacy branching model in the history

© ... ghts reserved.

Moving Perforce Data: Start Over

The steps involved in the start over process are:

Step 01

Extract the head of each branch in Perforce

Step 02

Check the data in the new empty Git repository

- It is fast and simple
- It can redesign branching model and workflow
- It provides Legacy Perforce server which is used for read-only access

ghts reserved.

Migration to Git from Perforce

The process involves eight steps:

- Moving Perforce data
- **Mapping users and permissions**
- Managing large binary files
- Working on complex dependencies
- Structuring the team during migration

- Map users and permissions into a new Bitbucket project.
- Less time is consumed if LDAP is used for a user directory. Otherwise, can extract a set of user accounts from Perforce.
- User accounts can be entered into Bitbucket taking one project at a time

©Simplilearn. All rights reserved.

Migration to Git from Perforce

The process involves eight steps:

- Moving Perforce data
- Mapping users and permissions
- **Managing large binary files**
- Working on complex dependencies
- Structuring the team during migration

If large binary blobs are stored in Perforce.

- Can use Git LFS
- Can use a regular artifact management system

©Simplilearn. All rights reserved.

Migration to Git from Perforce

The process involves eight steps:

- Moving Perforce data
- Mapping users and permissions
- Managing large binary files
- **Working on complex dependencies**
- Structuring the team during migration

- Working copy maps in read-only copies of data from several modules.
- It can be done using:
  - Submodules
  - Subtrees
  - CI/CD
  - Artifact management system

©Simplilearn. All rights reserved.

Migration to Git from Perforce

The process involves eight steps:

- Moving Perforce data
- Mapping users and permissions
- Managing large binary files
- Working on complex dependencies
- **Structuring the team during migration**

Consider a project plan during the migration phase:

- Migrate team-by-team and project-by-project
- Migrate incrementally
- Use Git Fusion to do a two-way data exchange
- Invest in communicating the changes to the team

©Simplilearn. All rights reserved.

Migration to Git from Perforce

The process involves eight steps:

Mirroring the data

Using ALM tools

Defining success

- It reduces the effect of latency
- It has a more complex system to run a set of local mirrors

©Simplilearn. All rights reserved.

Migration to Git from Perforce

The process involves eight steps:

Mirroring the data

↓

Using ALM tools

↓

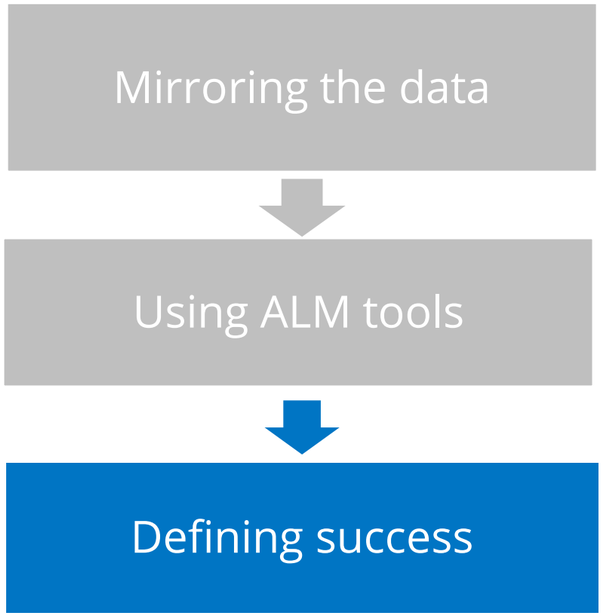
Defining success

- Receives lot of choices during transition from Perforce to Git
- Every developer and ALM tool work with Git

©Simplilearn. All rights reserved.

Migration to Git from Perforce

The process involves eight steps:



- The best approach for verification is to use CI/CD
- Check if all the test passes can still deploy the software and older builds can still pass through the CI/CD pipeline

©Simplilearn. All rights reserved.

Migration to Git from Perforce

Problem Statement: Your team wants the file to support distributed version control and is free of cost. So, you have to migrate to Git from Perforce for sharing the project files with your coworkers.

Steps to Perform:

1. Install Python and P4
2. Create a repository
3. Open Git Bash
4. Clone Perforce to Git
5. Add the GitHub repository
6. Push the code to remote GitHub

UNASSISTED PRACTICE

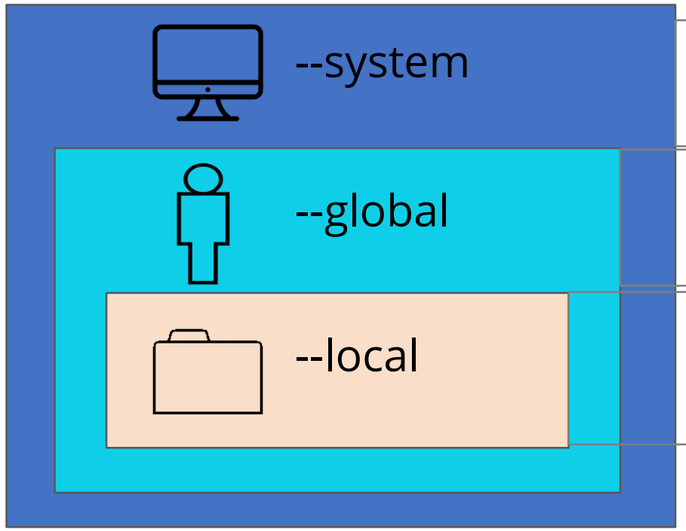
©Simplilearn. All rights reserved.

Overview of Git Configuration Level

©Simplilearn. All rights reserved.

Git Configuration Level

The git config command allows to configure the Git settings.



Command: '*git config –system*'
Saves to: */etc/gitconfig*

Command: '*git config –global*
Saves to: *~/.gitconfig* (On windows it will be users)

Command '*git config –local* (Or just 'git config')
Saves to: *.git/config*

File follows the same pattern

NOTE

Local overrides Global and Global overrides System Level.

©Simplilearn. All rights reserved.

Configure Git

Problem Statement: The configuration created during Git installation was lengthy. Your team wants Git to be configured for ease of use.

Steps to Perform:

1. Install Git
2. Configure the username and email ID

UNASSISTED PRACTICE

©Simplilearn. All rights reserved.

Overview of Basic Git Commands

©Simplilearn. All rights reserved.

Basic Git Commands

Below is the list of some basic Git commands:

Task	Explanation	Commands
Tell Git who you are	Configure the author's name and email address	git config --global user.name "Simplilearn" git config --global user.email simplilearn@example.com
Create a new local repository	Create a repository	git init
Check the repository	Create a working copy of a local repository	git clone /path/to/repository
Check the repository	Use a remote server	git clone username@host:/path/to/repository

©Simplilearn. All rights reserved.

Basic Git Commands

Below is the list of some basic Git commands:

Task	Explanation	Commands
Add files	Add one or more files to staging	git add <filename> git add *
Push	Send changes to the master branch	git push origin master
Commit	Commit changes to the head	git commit -m "Commit message"
Commit	Commit files added with **git add** and the files changed	git commit -a

©Simplilearn. All rights reserved.

Basic Git Commands

Below is the list of some basic Git commands:

Task	Explanation	Commands
Status	List the files that need to be changed, added, or committed	git status
Connect to a remote repository	Add the server to push for the connection	git remote add origin <server>
Connect to a remote repository	List all currently configured remote repositories	git remote -v
Search	Search the working directory for foo()	git grep "foo()"

©Simplilearn. All rights reserved.

Basic Git Commands

Below is the list of some basic Git commands:

Task	Explanation	Commands
Branches	Create a new branch and switch	git checkout -b <branchname>
Branches	Switch from one branch to another	git checkout <branchname>
Branches	List all the branches that tell you what branch you're currently in	git branch
Branches	Delete the feature branch	git branch -d <branchname>

©Simplilearn. All rights reserved.

Basic Git Commands

Below is the list of some basic Git commands:

Task	Explanation	Commands
Branches	Push the branch to your remote repository	git push origin <branchname>
Branches	Push all branches to your remote repository	git push --all origin
Branches	Delete a branch on your remote repository	git push origin :<branchname>

©Simplilearn. All rights reserved.

Basic Git Commands

Below is the list of some basic Git commands:

Task	Explanation	Commands
Update from the remote repository	Fetch and merge changes on the remote server	git pull
Update from the remote repository	Merge a different branch in an active branch	git merge <branchname>
Update from the remote repository	View all the merge conflicts	git diff
Update from the remote repository	View the conflicts against the base file	git diff --base <filename>
Update from the remote repository	Preview changes before merging	git diff <sourcebranch> <targetbranch>

©Simplilearn. All rights reserved.

Basic Git Commands

Below is the list of some basic Git commands:

Task	Explanation	Commands
Update from the remote repository	Manually resolve the conflicts and mark the changed file	git add <filename>
Tags	Use tagging to mark a significant changeset	git tag 1.0.0 <commitID>
Tags	Get the ID in use	git log
Tags	Push all tags to remote repository	git push --tags origin

©Simplilearn. All rights reserved.

Basic Git Commands

Below is the list of some basic Git commands:

Task	Explanation	Commands
Undo the local changes	Replace the changes in your working tree with the last content in head	git checkout -- <filename>
Undo the local changes	Fetch the latest history from the server and point to the local master branch	git fetch origin git reset --hard origin/master
Search	Search the working directory for foo()	git grep "foo()"

©Simplilearn. All rights reserved.

simplilearn

Create Git Alias

Problem Statement: You are facing issues with lengthy command lines. So you need to add alias for the commands.

Steps to Perform:

1. Install Git
2. Create an Alias

UNASSISTED PRACTICE

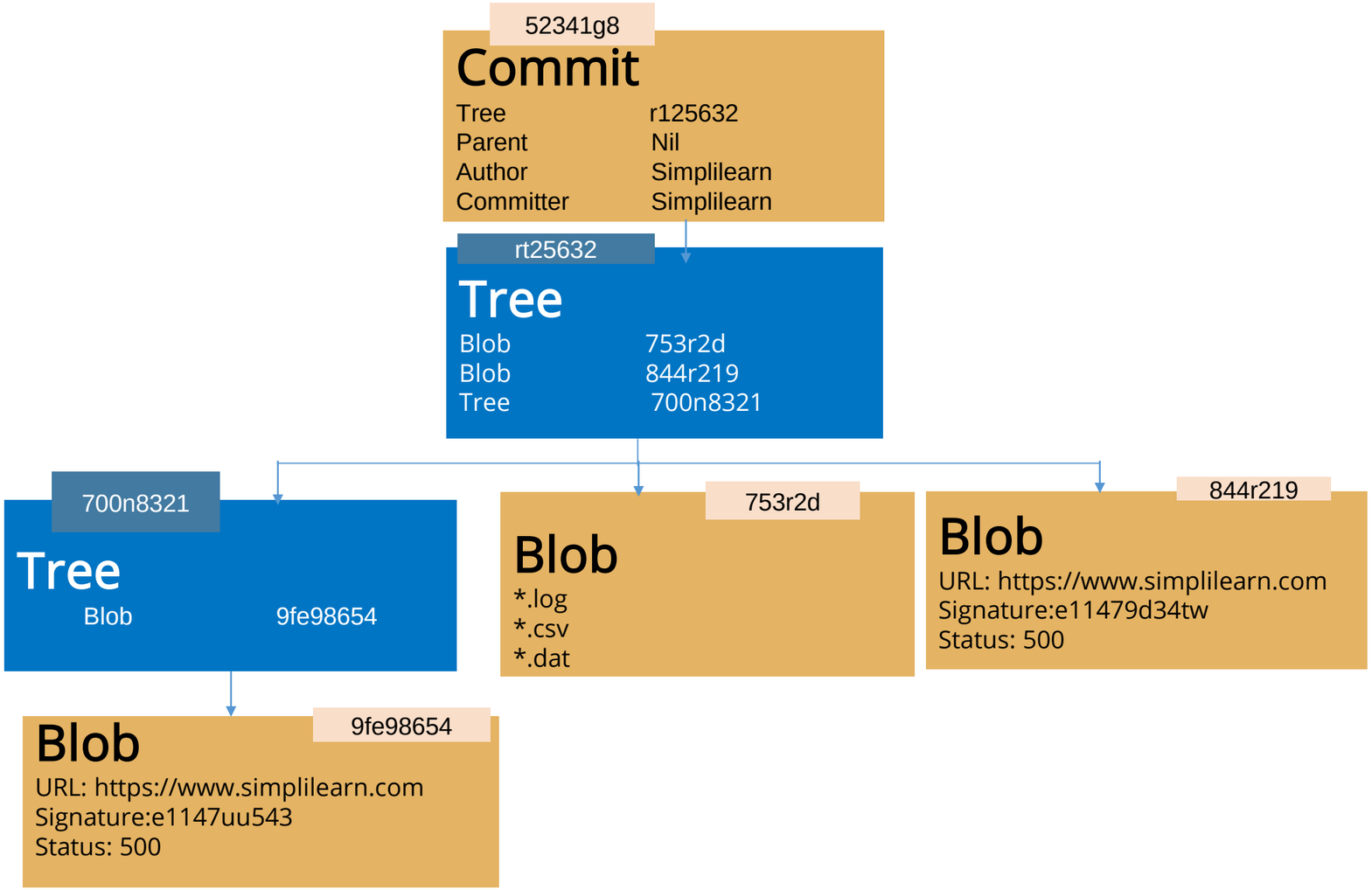
©Simplilearn. All rights reserved.

Overview of Web Scale Architecture

©Simplilearn. All rights reserved.

Web Scale Architecture

- Git uses a Directed Acyclic Graph (DAG) to represent history with commits.
- Git stores a snapshot of your repository whenever commit is created.



©Simplilearn. All rights reserved.

Difference between GitHub, Bitbucket, and GitLab

©Simplilearn. All rights reserved.

GitHub vs. GitLab vs. Bitbucket

	GitHub	GitLab	Bitbucket
Open Source	✓	✓	✗
CI pipeline	✓	✓	✗
Own APIs	✓	✓	✗
Git platform	✓	✓	✓
Active Bug tracking	✓	✓	✗
Supports	✓	✓	✓

©Simplilearn. All rights reserved.

Key Takeaways

- Version control is a system that record changes to a set of files.
- Git is a version control system for tracking changes in computer files.
- SVN and Perforce are centralized version control systems
- The git config command allows you to configure your Git settings.

©Simplilearn. All rights reserved.

Knowledge Check

©Simplilearn. All rights reserved.

Knowledge Check 1	What are the characteristics of a version control system?

a. Recording changes to a file or a set of files over time

b. Identifying who made the changes and when

c. Comparing and reverting to a previous state

d. All of the above

©Simplilearn. All rights reserved.

Knowledge Check 1	What are the characteristics of a version control system?

a. Recording changes to a file or a set of files over time

b. Identifying who made the changes and when

c. Comparing and reverting to a previous state

d. All of the above

The correct answer is **d**

A version control system allows you to record changes to a file or a set of files over time, identify who made the changes and when, and compare and revert to a previous state.

©Simplilearn. All rights reserved.

Knowledge Check 2	Which of the following would be true of how a distributed version control system is used?

a. Each developer would manually copy their files into a time-stamped directory

b. Every developer can check out project files from a central server, but would have to refer to the central server for project history details

c. Every developer would have a local copy of the entire repository including files and project history

d. Every developer would have a local copy of the entire repository including files but excluding project history

©Simplilearn. All rights reserved.

Knowledge Check 2	Which of the following would be true of how a distributed version control system is used?

a. Each developer would manually copy their files into a time-stamped directory

b. Every developer can check out project files from a central server, but would have to refer to the central server for project history details

c. Every developer would have a local copy of the entire repository including files and project history

d. Every developer would have a local copy of the entire repository including files but excluding project history

The correct answer is **c**

Distributed version control can be used by every developer by having a local copy of the entire repository including files and project history.

©Simplilearn. All rights reserved.

Knowledge Check 3	What are the three levels of configurations available in Git?

a. System, Global, and Local

b. System, Global, and User

c. Global, User, and Repository

d. User, System, and Global

©Simplilearn. All rights reserved.

Knowledge Check 3	What are the three levels of configurations available in Git?

a. System, Global, and Local

b. System, Global, and User

c. Global, User, and Repository

d. User, System, and Global

The correct answer is **a**

The three levels of configurations available in Git are System, Global, and Local.

©Simplilearn. All rights reserved.

Git Basics

Duration: 30 mins.

Problem Statement:
Create a repository which will cover concepts like installation, configuration, basic command, and alias name.

LESSON-END PROJECT

©Simplilearn. All rights reserved.