DevOps

Caltech | Center for Technology & Management Education

Post Graduate Program in DevOps

simplilearn

Version Control System

Learning Objectives

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

- List the preferred tools for Source Code Management
- Demonstrate the workflow of Git
- Illustrate GitHub as a Source Code Management tool
- Demonstrate branching, switching branches, and merging branches in Git

Introduction

Source Code Management

Source code management (SCM) is where the development teams share and collaborate among themselves.

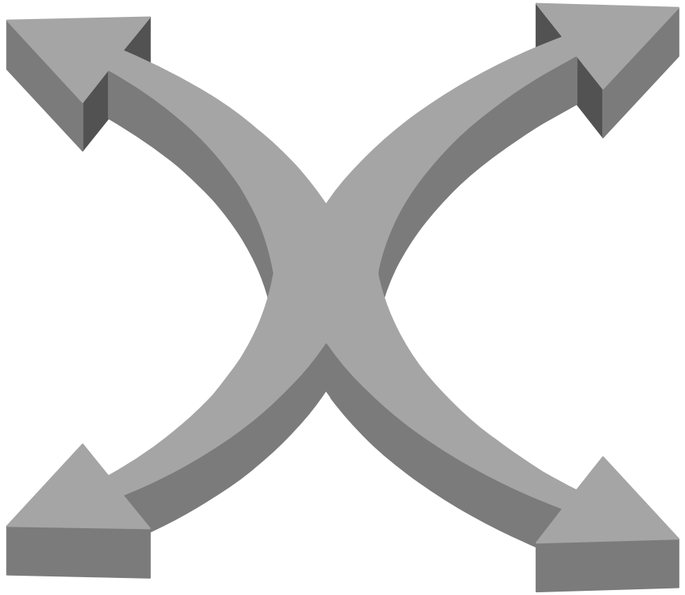
Purpose

- SCM is used to track modifications to a source code repository.
- It tracks a running history of changes to a code base and helps resolve conflicts when merging updates from multiple contributors.
- It is also synonymous with Version control.

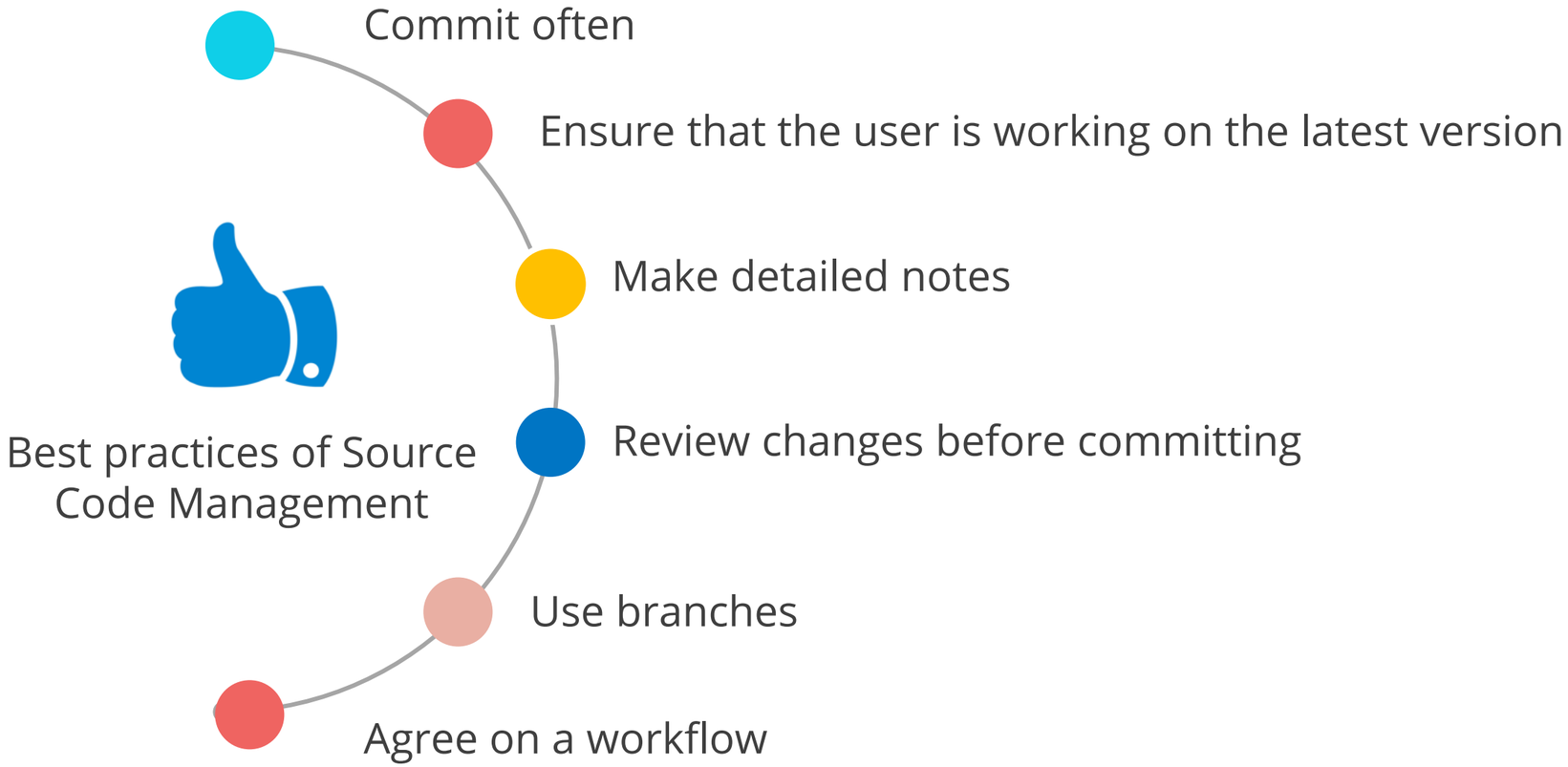
Source Code Management

These are some benefits of Source Code Management:

- Work together in teams
- Version history
- Generate release notes
- Backup of code



Source Code Management



Source Code Management: Tools

Some of the preferred tools for Source Code Management are:

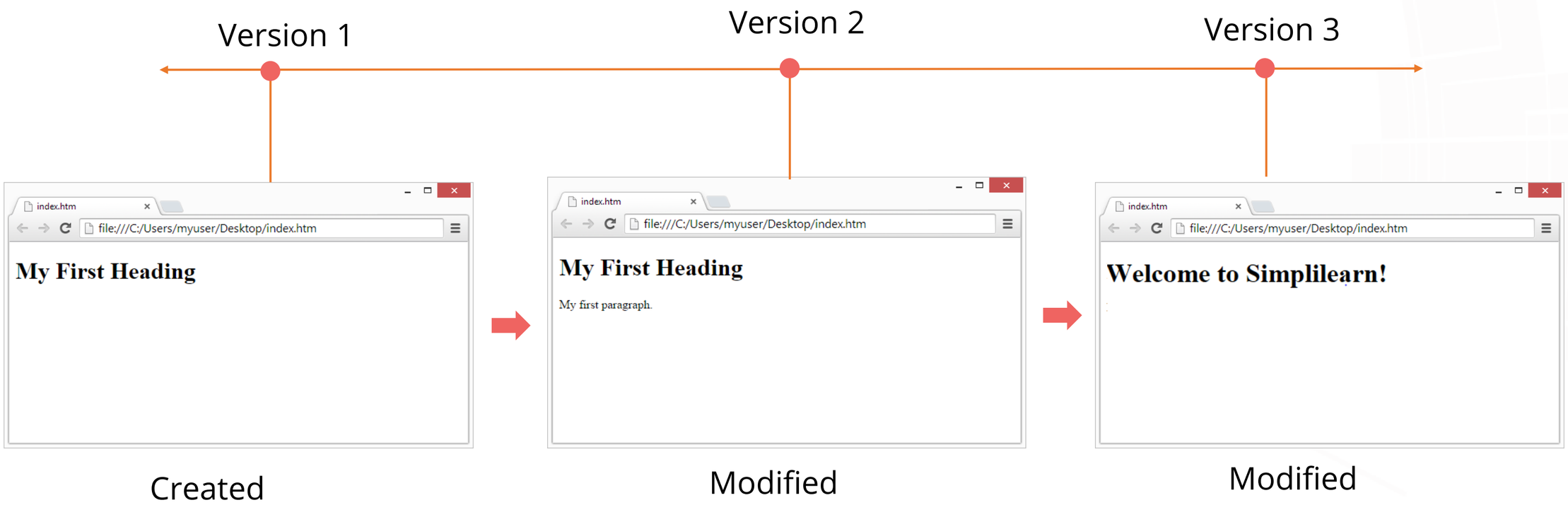
Caltech | Center for Technology & Management Education

simplilearn

Definition of Version Control Systems

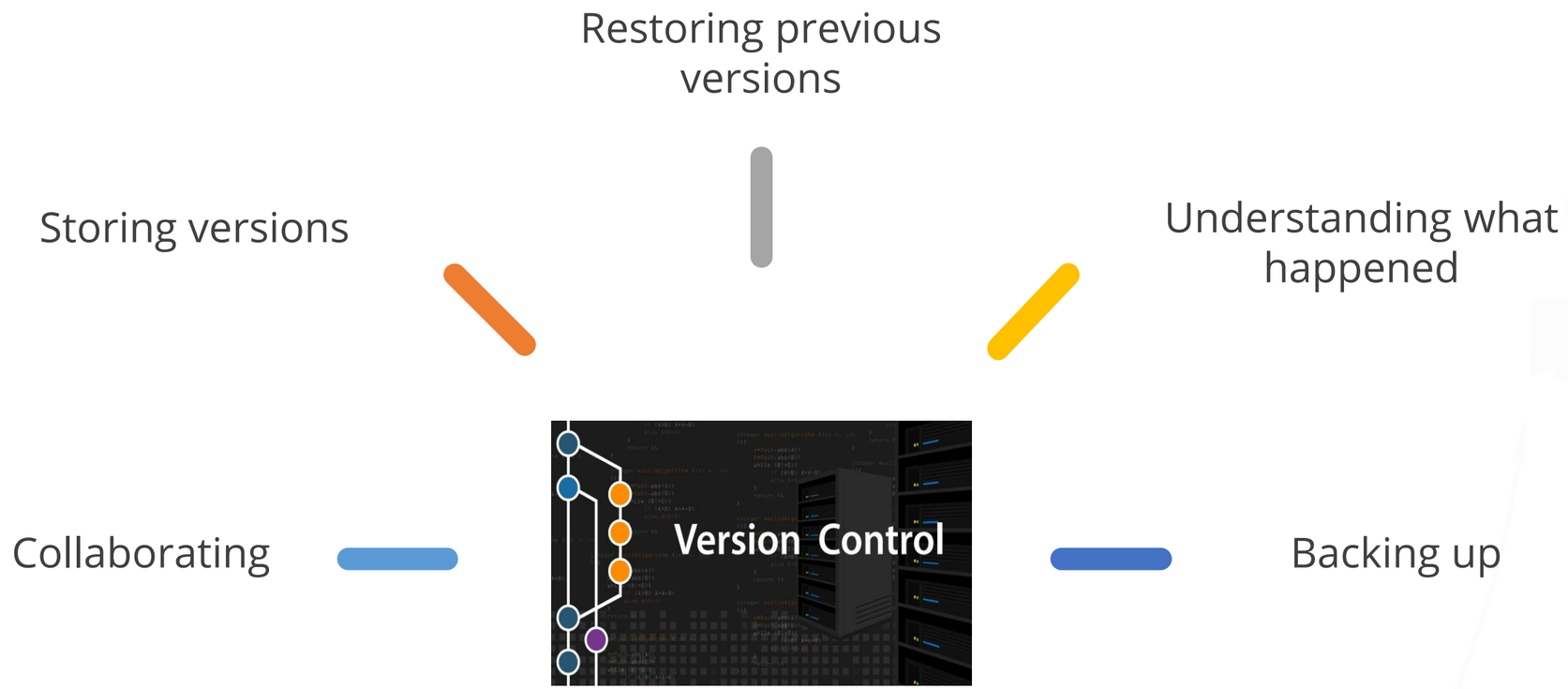
- Version control is a system that records changes to a set of files over a period of time to recall specific versions.
- Version Control System (VCS) can be used to store every version of an image or layout.

For example:



Benefits of Version Control System

Some benefits of version control system are:



What Is Git?

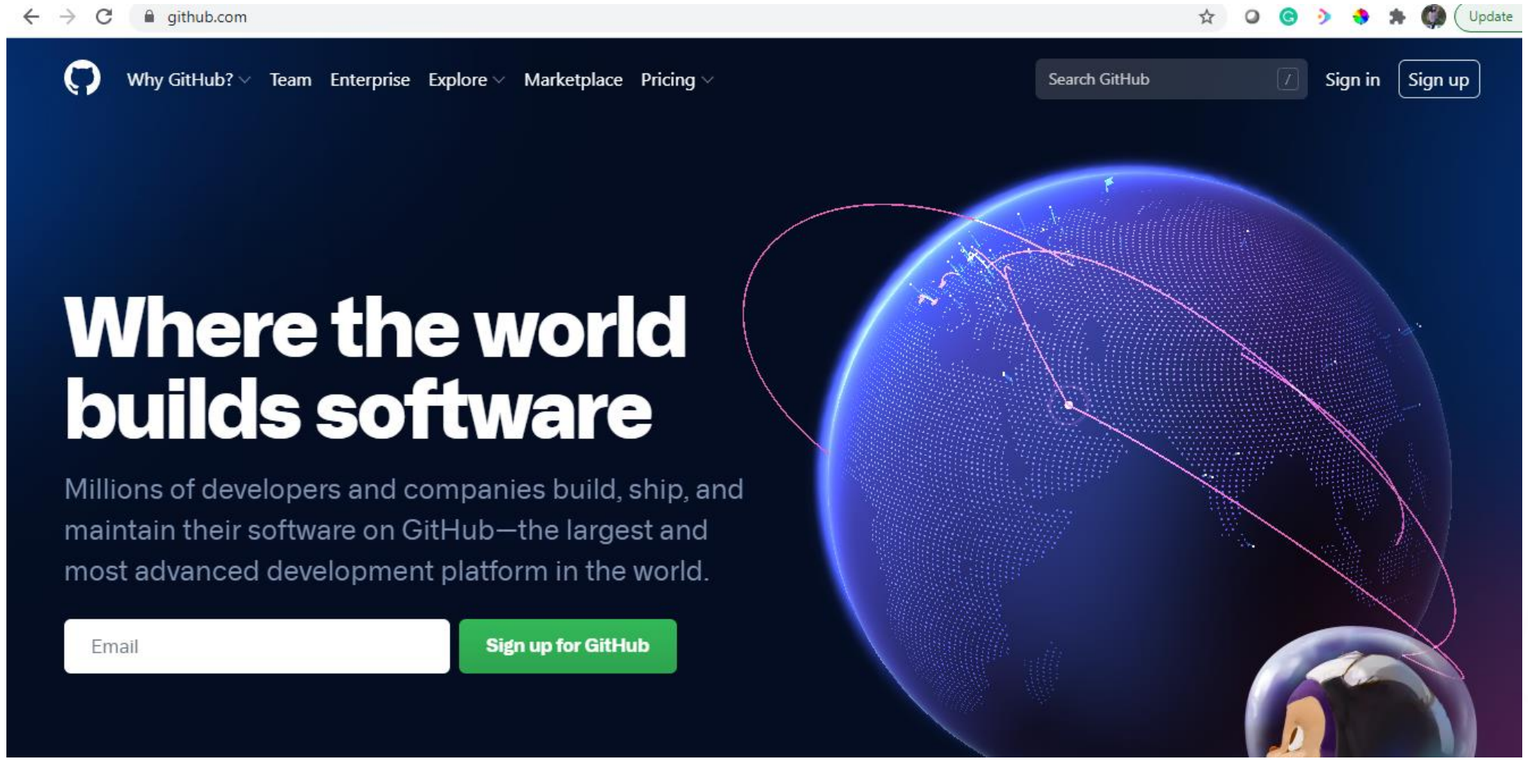
Git is a version control system for tracking changes in computer files. It is generally used for source code management in software development.

Developers

Git Server

What Is GitHub?

GitHub provides web-based git repository hosting service which provides web interface to upload files.



What Is Bitbucket?

Bitbucket is the Git repository management solution designed for professional teams. It provides a central place for managing git repositories, collaborate on source code, and guide the development process.

Features

- Access control to restrict access to the source code
- Workflow control to enforce a project or team workflow
- Pull requests with in-line commenting for collaboration on code review
- Jira integration for full development traceability
- Full Rest API to build features custom to the workflow if they are not already available from the Marketplace

What Is GitLab?

GitLab is a service that provides remote access to Git repositories.

Features

- Provides internal management of git repositories
- Keeps the user code private
- Deploys the change on the user code
- Provides user-friendly web interface layer

Working with Git

Git

Purpose of using Git:

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

Features of Git

- Compatible
- Distributed
- Non-linear
- Branched
- Lightweight

Features of Git

Fast

Open-source

Reliable

Secure

Economical

Advantages of Git

Has flexible environment

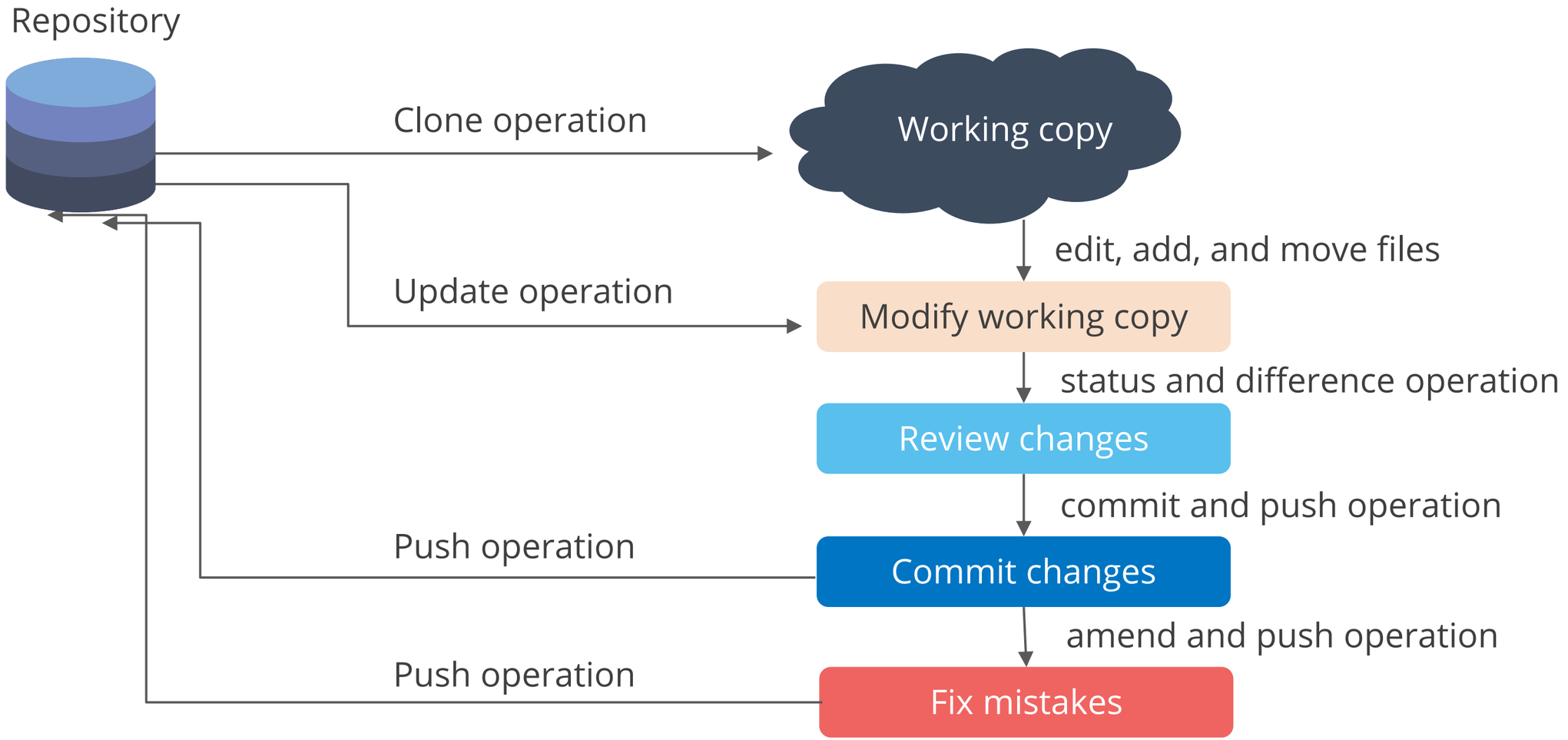
Keeps the files in a secure server

Creates and manages remote repositories

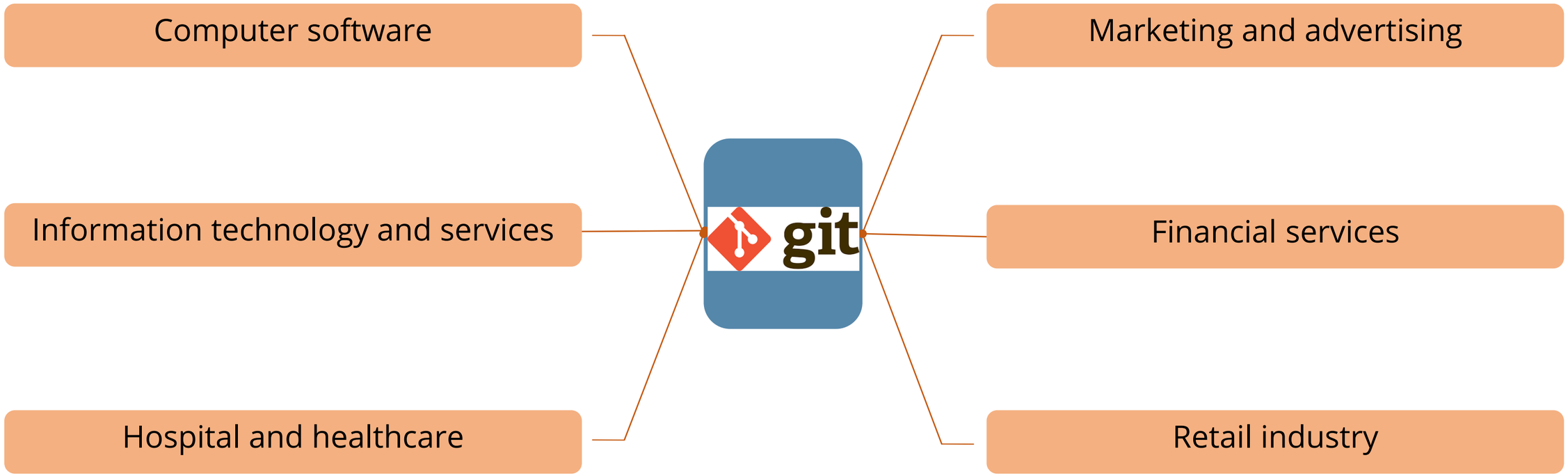
Facilitates branching and merging

Lifecycle of Git

The following diagram shows the lifecycle of Git:



Who Uses Git?



Basic Git Commands

Below is the list of some basic Git commands:

Task	Explanation	Commands
Tell Git who the user is	Configure the author's name and email address	git config --global user.name “Simplilearn" git config --global user.emailsimplilearn@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

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

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()"

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>

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
Branches	Delete a branch on your remote repository	git push origin :<branchname>

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>

Unassisted Practice

Install Git on Linux

Duration: 05 Min.

Problem Statement:

You are given a project to install Git on Linux platform and verify the installation.

Unassisted Practice: Guidelines

Steps to install Git on Linux :

1. Install Git.
2. Verify the installation.

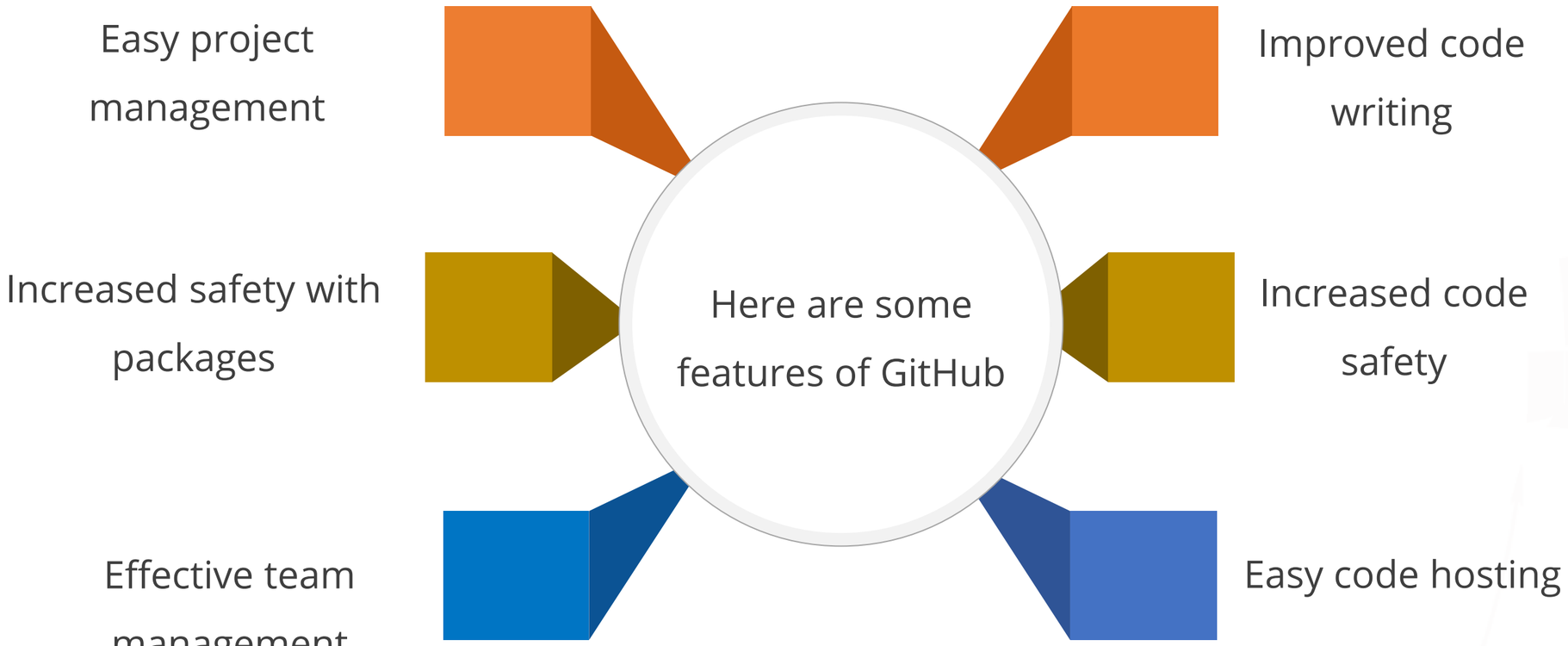
GitHub as an SCM Tool

GitHub

Purpose of using GitHub:

- Allows developers to share their code and view other's code.
- Helps to follow each other and rate each other's work
- Allows user to receive updates about the different projects which they are interested in and communicate with other members in public or private

Features of GitHub



Who Uses GitHub?

GitHub can be used by:

Individual

Community

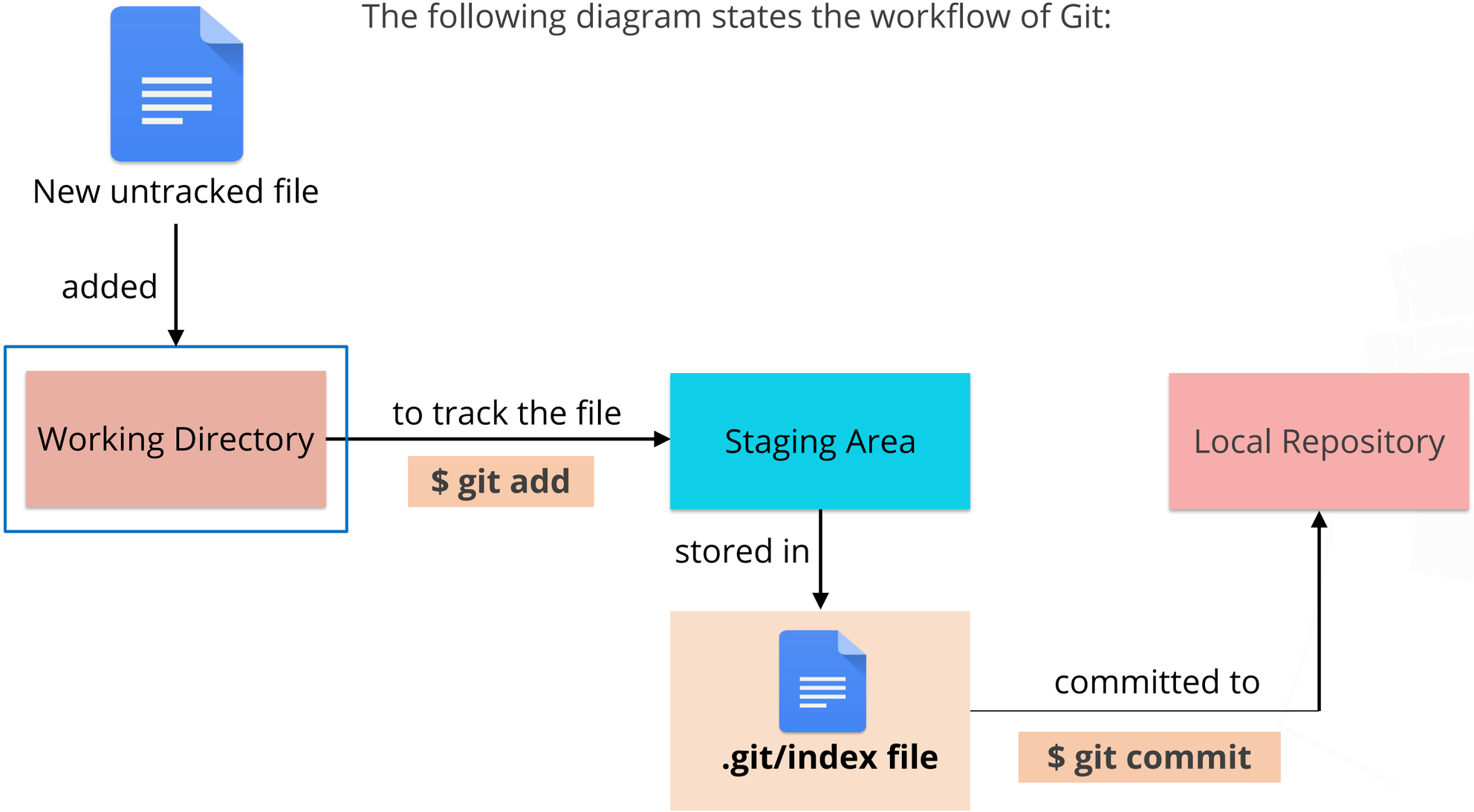
Business

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 provides web-based git repository hosting service which provides web interface to upload files.

Git Workflow

The following diagram states the workflow of Git:



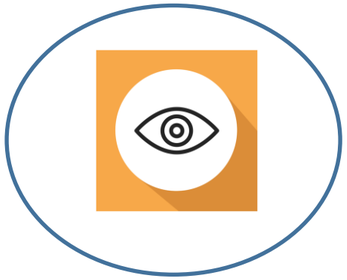
Characteristics of Git Repository

Can access files

Each repository belongs to a user or team

Project code can consist of single or multiple repositories

Can view public repositories

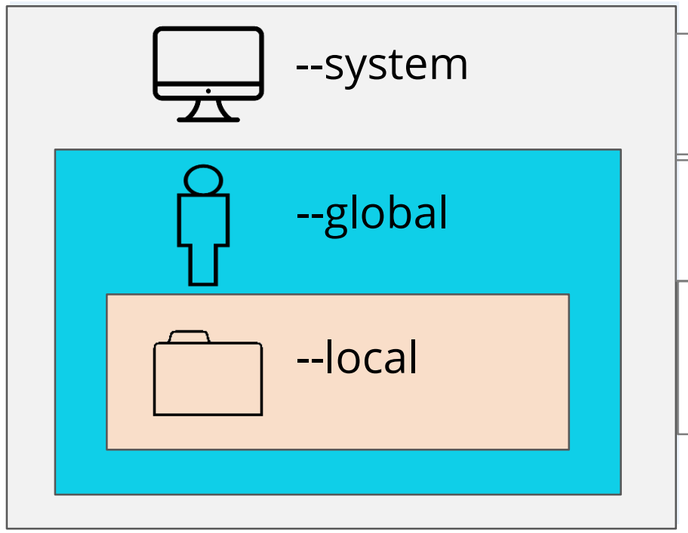


Only an admin can delete repositories

Each repository has a size limit of 2 GB

Git Configuration Level

The git config command allows to configure the Git settings.



Command: *'git config --system'*
Command: *'git config --global'*
Command: *'git config --local'*

NOTE

Local overrides Global and Global overrides System Level.

Assisted Practice

Create and Clone a GitHub Repository

Duration: 20 Min.

Problem Statement:

You are given a project to create a GitHub repository. Share the project files with your coworkers and clone the GitHub repository shared by your coworker to access the project files.

Assisted Practice: Guidelines

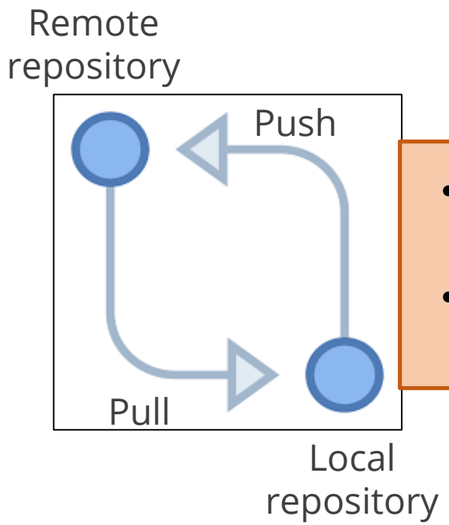
Steps to create and clone a GitHub repository :

1. Create a new github repository.
2. Edit the README file.
3. Upload a file to the repository.
4. Clone the github repository.

Fork, Push, and Pull in Git

Fork, Push, and Pull in Git

Fork is a copy of a repository. Forking a repository allows you to freely modify with changes without affecting the original project.



- Push helps the content to upload from local repository to remote repository.
- Pull helps the content to download from remote repository to local repository.

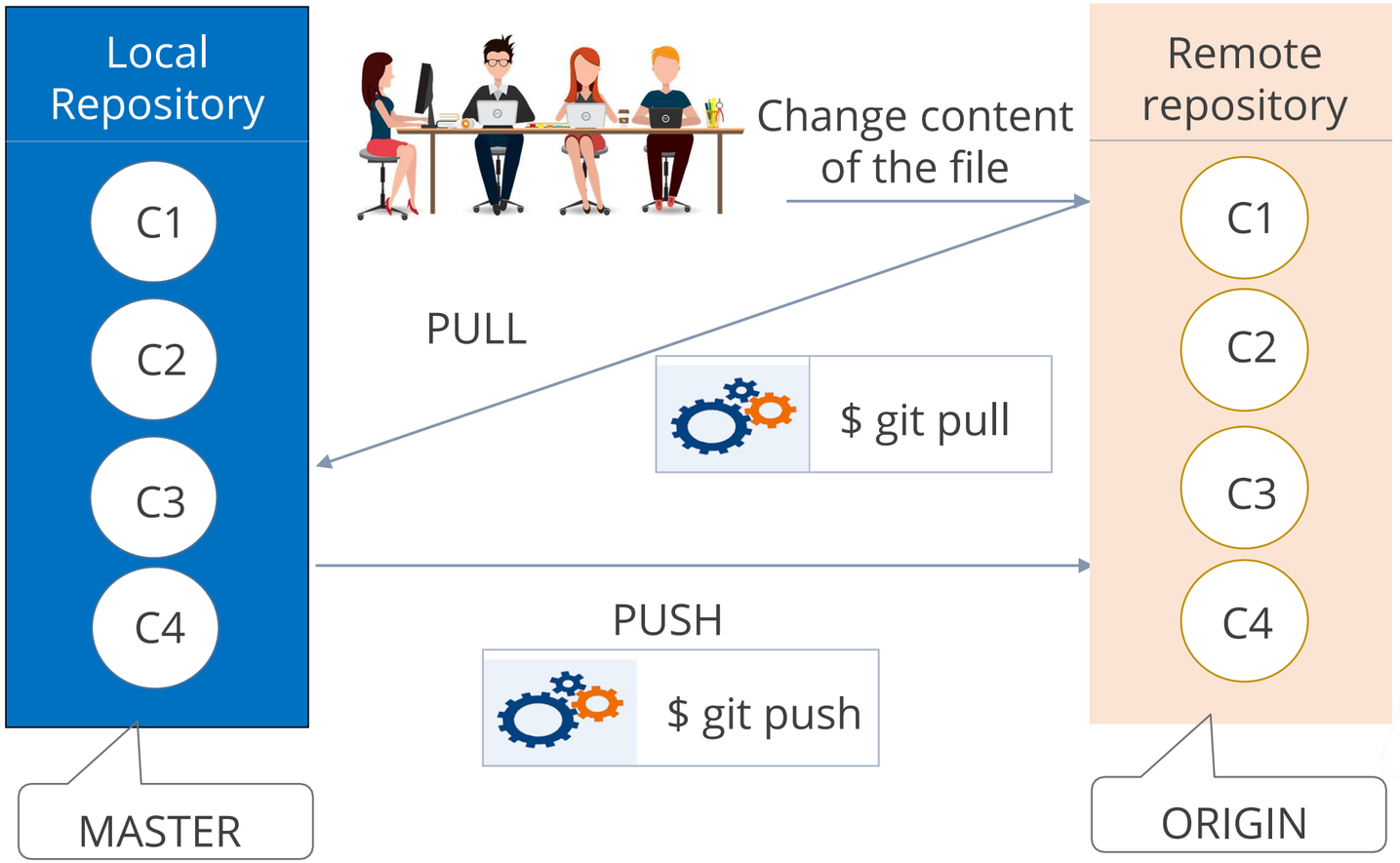
Fork, Push, and Pull in Git

The steps to create a repository in GitHub using fork and pull requests are:

1. Create a fork
2. Clone your fork
3. Modify the code
4. Push your changes
5. Create a pull request

Pushing and Pulling in Git

The diagram shows the workflow of push and pull in Git:



Assisted Practice

Create a Pull Request in Git

Duration: 20 Min.

Problem Statement:

You are given a project to experiment freely on your coworker's project without affecting the original project. After you modify the project, you have to create a pull request.

Assisted Practice: Guidelines

Steps to create a pull request in Git:

1. Create a fork.
2. Clone your fork.
3. Sync fork with the original repository.
4. Push your changes.
5. Create a pull request.

Assisted Practice

Push file to GitHub Repository

Duration: 20 Min.

Problem Statement:

You are given a project to create a file in your local repository and then push the changes from the local repository to the GitHub repository

Assisted Practice: Guidelines

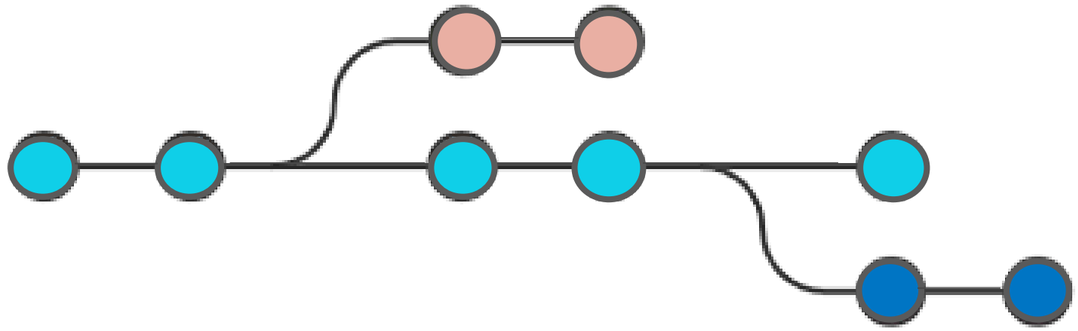
Steps to push file to GitHub repository:

1. Create a GitHub repository.
2. Create a repository on the local machine.
3. Push the changes in the local repository to GitHub.
4. Check the status of the local and remote repository.

Branching in Git

Introduction to Branches

A Git branch is a lightweight movable pointer to the commits.
It is a separate workspace, usually created when you need to experiment or test something, without impacting the main code base.



Purpose of Branches in Git

Purpose

- Branches give the users freedom to independently work on different modules and merge the modules when they finish developing them.
- Git branches are swift to be created and destroyed.
- They are cheap, considering the size they take.
- Branches in Git help the teams, which are in different parts of the world, work independently on independent features that would ultimately combine to produce a great project.
- The branches are very flexible.

Different Operations in Branching

Operations	Description
Create a branch	The first step in the process is to create a branch. The user can start with the default branch or create a new branch.
Merge a branch	Every branch in the Git repository may be merged with an already running branch. Merging a branch can help when the user is finished with the branch and wants the code to integrate into another branch code.
Delete a branch	The user can remove an existing branch from the Git repository. When the branch has completed its task, that is, it has already been merged, or the user no longer needs it in the repository for some reason, you may delete it.
Checkout a branch	A user can pull or checkout a branch that is already running to create a clone of the branch so that they can work on it.

Branch Commands

$ git branch <branch name>	Creates a new branch
$ git branch	Lists all branches in the current repository
$ git checkout <branch name>	Switches to branches
$ git merge <branch name>	Merges branches

NOTE

To create a new branch and switch to it, execute: $ git checkout -b<branch-name>

Assisted Practice

Create a Branch in Git

Duration: 15 Min.

Problem Statement:

You are working on a project and have a couple of commits already on the master branch. You've decided that you're going to work on a new issue in a new branch, so you must create a new branch for that.

Assisted Practice: Guidelines

Steps to create a branch in Git:

1. Create a new repository.
2. Clone the GitHub repository.
3. List all the branches in your repository.
4. Create a new branch.
5. Verify the creation of the new branch.
6. Rename an existing branch.
7. Delete the branch.
8. Verify the deletion of the branch.

Switching Branches in Git

Switching Branches in Git

The **switch** command allows you to switch your current HEAD branch. It's relatively new and provides a simpler alternative to the classic **checkout** command.

The **git checkout** command allows you to switch branches by updating the files in the working tree to match the version stored in the branch that the user wishes to switch to.

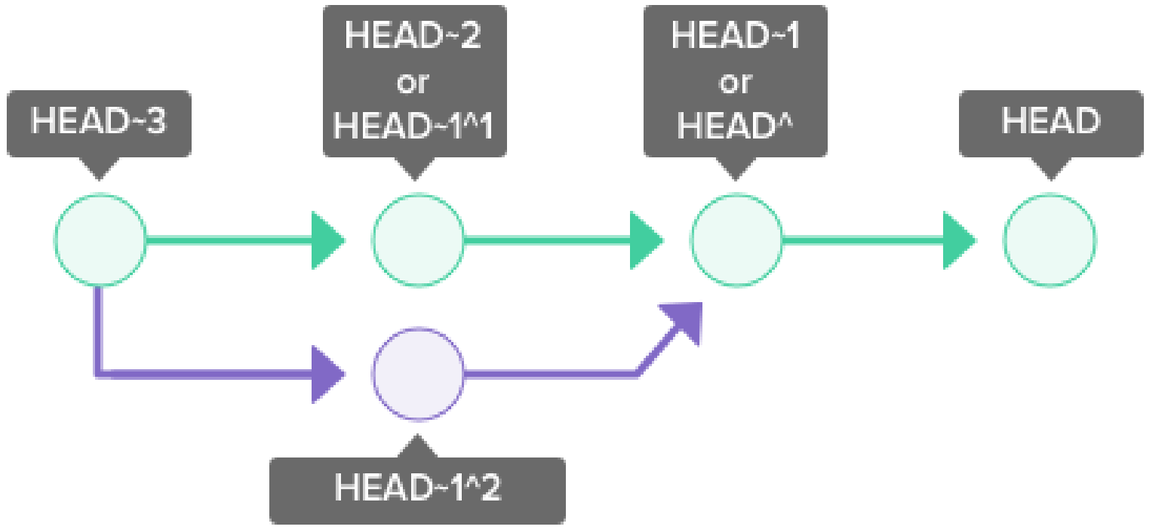
Switching Branches in Git

Git Operations	git checkout	git switch
To switch from one branch to another	git checkout <branchname>	git switch <branchname>
Creates a new branch and also switches to it	git checkout -b <branchname>	git switch -c <branchname>

Switching Branches in Git

Git HEAD

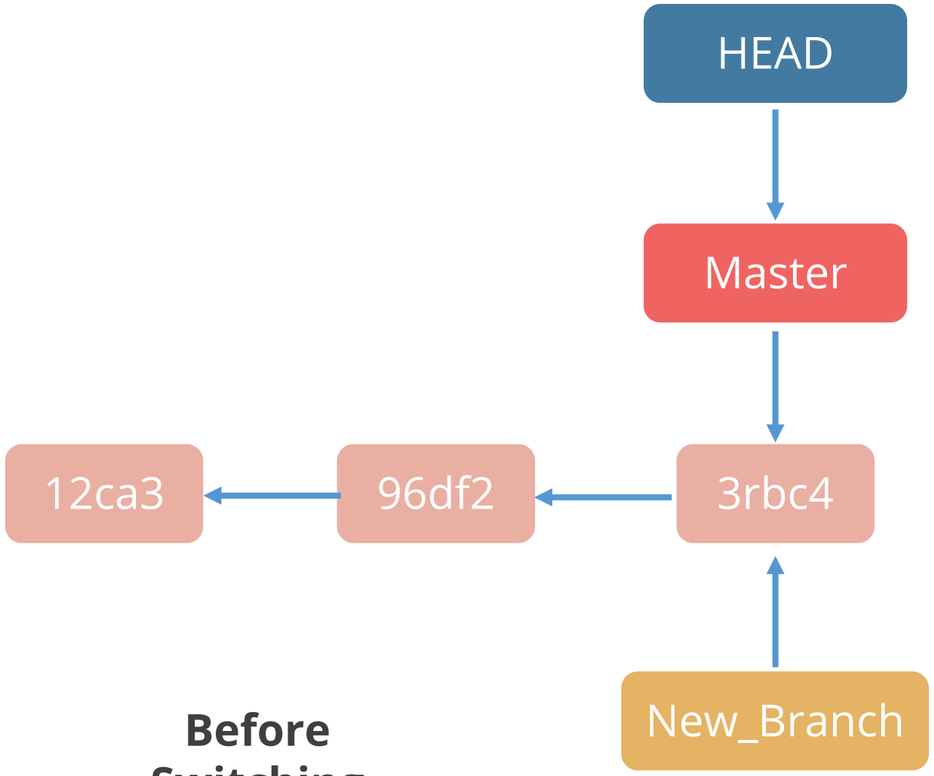
HEAD is used to represent the current snapshot of a branch. For a new repository, Git will by default point HEAD to the master branch. Changing where HEAD is pointing will update your current active branch.



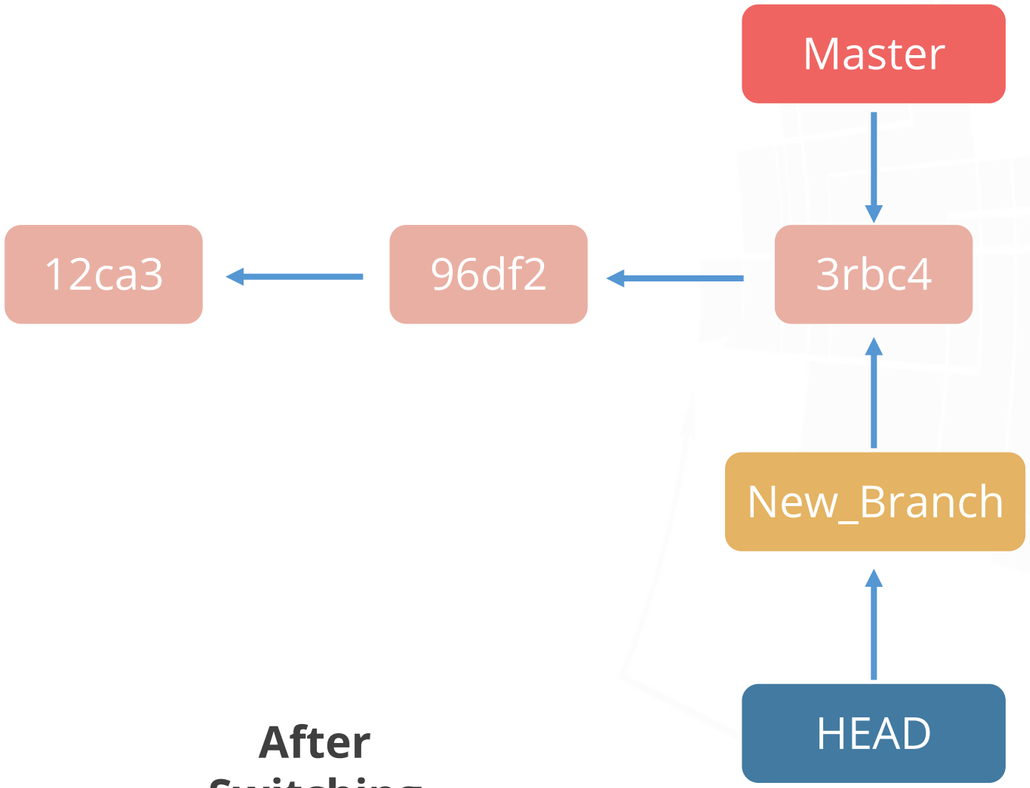
Switching Branches in Git

Git checkout <existing branch> can be used to switch to an existing branch.

Git checkout New_Branch



Before Switching



After Switching

Caltech | Center for Technology & Management Education

simplilearn

Assisted Practice

Switching Branches in Git

Duration: 15 Min.

Problem Statement:

You are given a project and while working in your local repository, you wish to checkout and work on branch code rather than the main code line. You have to switch over to the new branch and add commit to it.

Assisted Practice: Guidelines

Steps to switching branches in Git:

1. Create a new branch.
2. Switch to the new branch.
3. Create a file and commit the changes.
4. Check the status of the new branch.
5. Switch back to the main branch.

Merging Branches in Git

Merging Branches in Git

Merging is Git's way of putting a forked history back together again.
The **git merge** command takes the independent lines of development created by git branch and integrates them into a single branch.

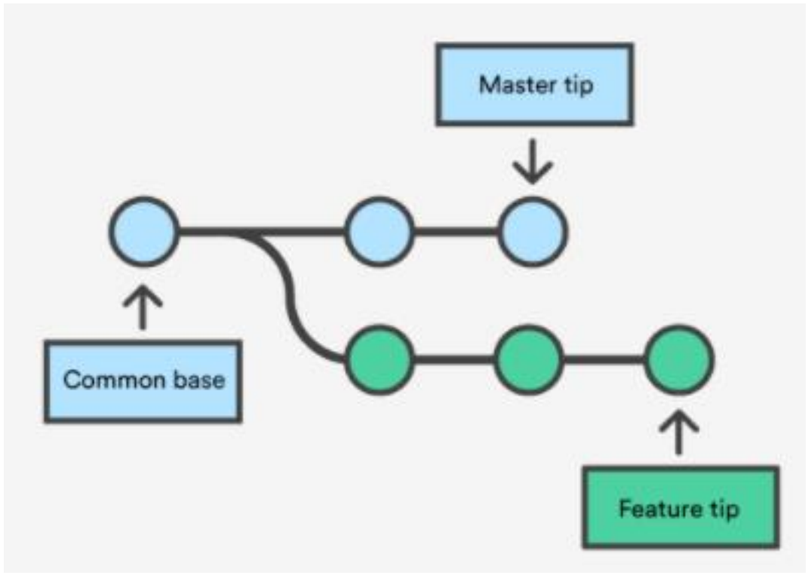
Steps to merge branches:

Switch to the branch you want to merge

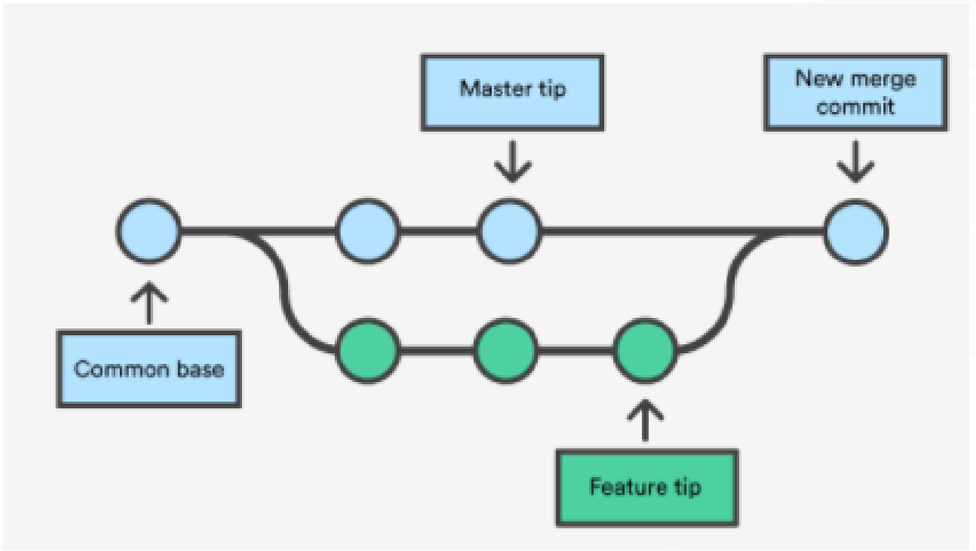
Execute: **$ git merge <branch name>**

Merging Branches in Git

Git merge will combine multiple sequences of commits into one unified history. In most of the cases, git merge is used to combine two branches.



Before Merging



After Merging

Assisted Practice

Merging Branches in Git

Duration: 15 Min.

Problem Statement:

You are given a project. You have completed working on the new merge and you are ready to merge into your master branch.

Assisted Practice: Guidelines

Steps to merge a branch in Git:

1. Create a new branch.
2. Create a new file in the new branch.
3. Switch to the main branch.
4. Merge the branches.
5. Push the changes to the remote repository.

Key Takeaways

- Software configuration management (SCM) is a set of processes, policies, and tools that organizes the development process.
- Git is a version control system for tracking changes in computer files.
- GitHub provides web-based git repository hosting service which provides web interface to upload files.
- The different operations of branching are create a branch, merge a branch, delete a branch, and checkout a branch.

Lesson-End Project

Create a New Branch and Merge the Branch in Git

Project Agenda: To create a new branch and merge it in Git

Description: You are working in a IT company. Your company is working on a project which contains three modules and you have been asked to work on one of the module. Your company asked you to upload all the project files in the GitHub repository. You can create a new branch and work on it so that you don't impact the main code base. After you complete your module you can merge it into the master branch

Thank You