git

Content

- git Overview
- git Installation
- git Account Setup
- git integration with IDE
- git init
- git clone
- git add
- git commit
- git push
- git pull

Overview

- Git is a distributed version control system for software development.
- It helps manage and keep track of changes to the source code of a project over time, and supports collaboration among multiple contributors.
- Git allows you to revert to previous versions of your code, collaborate with other developers on a project, and deploy your code in a more organized and efficient manner.

Installing Git

Download git:

https://git-scm.com/downloads

Github Account

- Download git:

  https://github.com/login

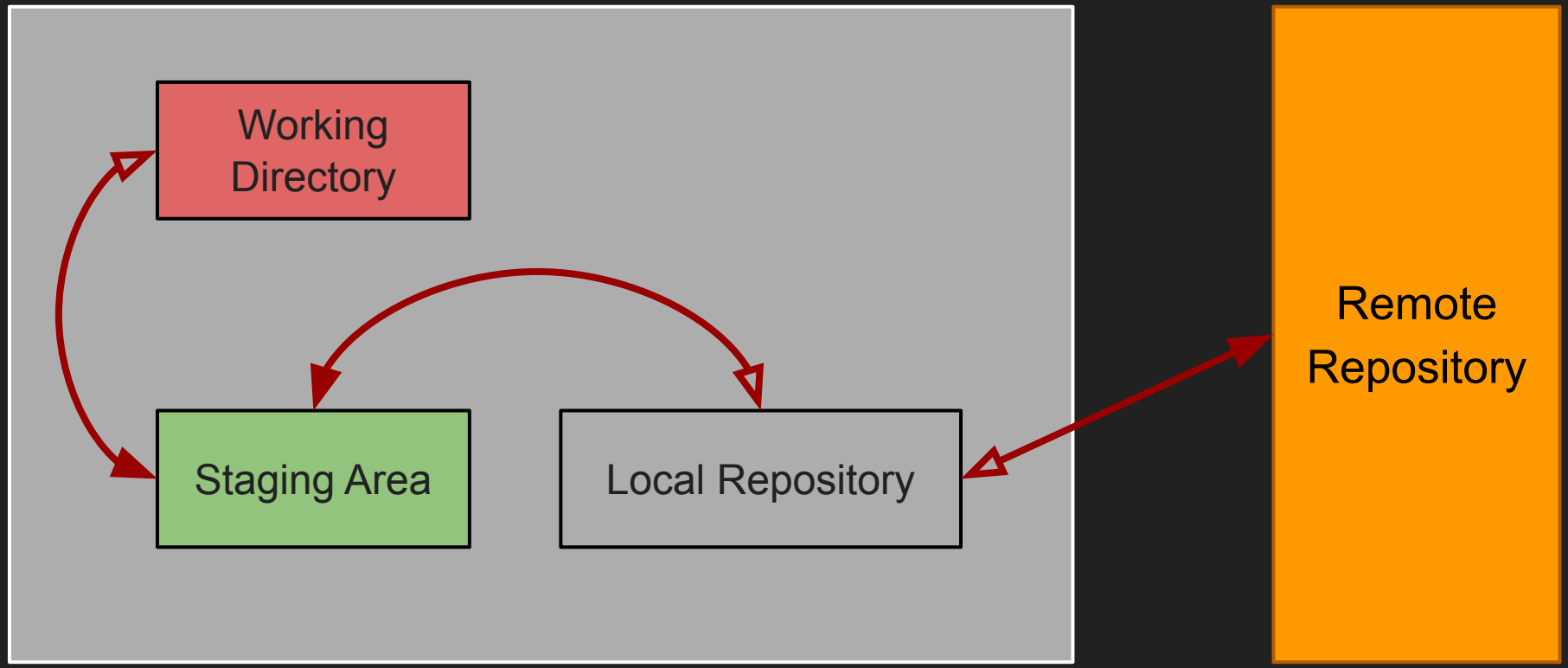
- Create Repository
- Clone repository

Git architecture consists of several components

- **Repositories**: A Git repository is a central location where all the versioned files and their history are stored. It can be either local (on a developer's machine) or remote (hosted on a server).
- **Objects**: Git uses a database of objects to store the contents of files, commits, and other metadata. There are four types of objects in Git: blobs (for file contents), trees (for directory structures), commits, and tags.
- **References**: References are pointers to specific Git objects, such as a particular commit. A branch is a reference that moves as new commits are made.
- **The staging area**: The staging area (also known as the index) is a temporary holding area where changes to files are stored before they are committed to the repository.
- **The working tree**: The working tree is the set of files in a Git repository that are checked out and available for editing.

These components work together to allow multiple users to make changes to the same codebase, keep track of their changes, and merge their changes back together when necessary.

Git architecture consists of several components



Configuring Git

```
git config --global user.name "John Doe"

git config --global user.email johndoe@example.com
```

git init

- **git init** is a command that is used to initialize a new Git repository. This command creates a new, empty Git repository in the current directory, which allows you to start version-controlling your files.
- After running git init, you can start tracking changes to your files by adding them to the repository and committing changes to the repository's history.

git status

- git status is a Git command that displays the current state of a Git repository.
- The git status command shows you which files have been modified, staged, or are new and not yet tracked by Git.
- It also provides information about the current branch and the status of the repository's connection to a remote repository.
- Running git status is a useful way to get a quick overview of what has changed in your repository and what you need to do next (e.g., stage changes, make additional modifications, or push changes to a remote repository).
- The output of git status is color-coded, making it easy to quickly identify the status of each file in the repository.

git clone

- git clone is a Git command that allows you to copy an existing Git repository from a remote source (such as a server or a colleague's repository) onto your local machine.
- The git clone command creates a complete local copy of the repository, including all of its history and branches, and sets up a remote connection to the original repository so that you can easily keep your local copy in sync with the remote repository.
- Running git clone is the most common way to start using a Git repository that you did not create yourself.

git add

- git add is a Git command that is used to stage changes in a Git repository.
- The git add command allows you to selectively choose which changes you want to include in the next commit, giving you fine-grained control over the repository's history.
- When you make changes to a file in a Git repository, those changes are not automatically included in the next commit. Instead, you must use git add to tell Git which changes you want to include.
- When you run git add, Git stages the changes, which means that it marks the changes as ready to be committed.
- Once you have staged the changes you want to include in a commit, you can run git commit to save the changes to the repository's history.

git add

git add file1.txt file2.txt

This command is for adding specific files.

git add .

This command is for adding all the files and sub folders in the current directory.

git add --all (git add -A)

This command is for adding all the files and folder in the entire repository

git commit

- git commit is a Git command that allows you to save changes to a Git repository.
- The git commit command creates a new "commit" object in the repository's history that represents a set of changes made to the repository's files.
- A commit contains a reference to the state of the repository's files at the time the commit was created, as well as a commit message that provides a brief description of the changes that were made.
- Commits serve as snapshots of the repository's history, allowing you to easily track and revert changes to your code over time.
- To create a new commit, you must first stage the changes you want to include in the commit using the git add command, and then run git commit with a commit message to finalize the commit.

git push

- git push is a Git command that is used to upload local changes to a remote Git repository.
- The git push command allows you to take the changes you have made in your local repository and share them with others by pushing them to a remote repository.
- When you run git push, Git transfers the committed changes from your local repository to the remote repository, updating the remote repository with your latest changes.
- The git push command requires a remote repository to be specified, and also requires that you have write access to the remote repository.
- The git push command is typically used to collaborate with others on a project, allowing you to share changes and work together on a common codebase.

git branch

git branch is a Git command that allows you to create, list, or delete branches in a Git repository. A branch in Git is a separate line of development that allows you to work on a feature or bug fix without affecting the main branch of the repository (typically referred to as the "master" branch).

Here are some common uses of git branch:

- To list all branches in a repository: git branch
- To create a new branch: git branch <branch-name>
- To switch to an existing branch: git checkout <branch-name>
- To delete a branch: git branch -d <branch-name>
- To create a new branch and switch to it in one command: git checkout -b <branch-name>

Branches are a powerful tool for managing and organizing your work in Git. By creating separate branches for different features or bug fixes, you can work on them in parallel, and then merge them into the main branch when they are ready. This allows you to isolate your work and collaborate more effectively with other contributors to the repository.

git merge

git merge is a Git command that allows you to combine multiple branches into a single branch. The git merge command integrates the changes from one branch into another branch, resulting in a unified history of both branches.

The most common use of git merge is to integrate changes from a feature branch into the main branch (typically referred to as the "master" branch) of a repository. To do this, you switch to the branch that you want to merge into (the main branch), and then run the git merge command, followed by the name of the branch that you want to merge from.

$ git merge <from-branch-name>

git fetch

git fetch is a Git command that retrieves updates from a remote repository, but does not integrate them with your local repository. The git fetch command downloads the latest version of a remote branch, but does not merge it into your local branch. Instead, it stores the remote branch as a separate branch in your local repository, which you can inspect or merge later.

git fetch is commonly used to synchronize your local repository with a remote repository, such as a remote branch hosted on a code-sharing platform like GitHub or GitLab. By fetching the latest changes from the remote repository, you can see what has changed and decide whether to merge the changes into your local repository.

git pull

git pull is a Git command that retrieves updates from a remote repository and integrates them with your local repository.

The git pull command combines the functionality of two other Git commands: git fetch, which downloads updates from a remote repository, and git merge, which integrates the updates into your local repository.

When you run git pull, Git retrieves updates from the remote repository, and automatically merges them into the branch you are currently working on.