**Git 29-8**

## ****Source Code Management****

## Git

Git is Management tool to store in source in Git...

Git is a modern and widely used **distributed version control** system in the world.

Git via the command line as well as with [GitHub](https://www.javatpoint.com/what-is-github).

## What is Git

**Git** is an **open-source distributed version control system**. It is designed to handle minor to major projects with high speed and efficiency. It is developed to co-ordinate the work among the developers

Git is foundation of many services like **GitHub** and **GitLab**, but we can use Git without using any other Git services. Git can be used **privately** and **publicly**.

Git was created by **Linus Torvalds** in **2005** to develop Linux Kernel. It is also used as an important distributed version-control tool for **the DevOps**.

Git is easy to learn, and has fast performance

## Features of Git



## Benefits of Git



# How to Install Git on Windows

To download the Git installer, visit the Git's official site and go to download page. The link for the download page is <https://git-scm.com/downloads>

## Install Git:

Select **yes** to continue.

After the selecting **yes** the installation begins, and the screen

Click on **next** to continue.

Default components are automatically selected in this step. You can also choose your required part

Click on **next** to continue.

The default Git command-line options are selected automatically. You can choose your preferred choice.

Click **next** to continue.

The default transport backend options are selected in this step.

Click **next** to continue.

Select your required line ending option and

click **next** to continue.

Select preferred terminal emulator

clicks on the **next** to continue.

This is the last step that provides some extra features like system caching, credential management and symbolic link. Select the required features

click on the **next** option.

The files are being extracted in this step.

**The Git installation is completed.**

The **Git Gui** looks like as

**Git is a distributed version control system (VCS)** used for tracking changes in source code during software development. It was created by Linus Torvalds in 2005 to manage the development of the Linux kernel. Git is designed to be fast, efficient, and flexible, making it a popular choice for managing version history, collaborating on code, and coordinating the work of multiple developers on a project.

**Repository (Repo):** A repository is a directory or storage space where your project's files and version history are stored. It contains all the files, commits, branches, and other data related to your project.

**Commit**: A commit is a snapshot of the changes you've made to your code at a specific point in time. Each commit is accompanied by a message that describes the changes you've made.

**Branch:** A branch is a separate line of development within a repository. Branches allow multiple developers to work on different features or fixes simultaneously without interfering with each other's work. The main branch is typically called "master" or "main."

**Merge**: Merging is the process of integrating changes from one branch into another. It's used to combine the work done in different branches and create a unified history.

**Pull Request (PR):** In collaborative environments like GitHub or GitLab, a pull request is a request to merge changes from one branch into another. It allows for code review and discussion before changes are merged.

***Clone***: Cloning is the process of creating a copy of a repository on your local machine. This copy includes all the version history and branches, allowing you to work on the project locally.

**Remote**: A remote is a version of the repository that is hosted on a server, like GitHub, GitLab, or Bitbucket. It provides a centralized location for collaborating and sharing code with others.

**Push**: Pushing refers to sending your committed changes from your local repository to a remote repository. This makes your changes available to others and updates the remote repository's history.

**Pull**: Pulling is the process of fetching changes from a remote repository and integrating them into your local repository. It's used to stay up-to-date with the latest changes made by others.

**Version Control**: Version control is the practice of tracking and managing changes to files over time. It allows developers to collaborate effectively, revert to previous states, and maintain a clear history of changes.