Understanding the basics of version control systems (VCS)

Q: What is a version control system (VCS)?

A: A VCS is a tool that helps manage changes to source code or other collections of information over time. It allows multiple people to work on a project simultaneously, tracks changes, and maintains a history of revisions.

Q: What are the main benefits of using a VCS?

A: The main benefits include collaboration, history tracking, branching and merging, backup and restore, and code review capabilities.

Introduction to Git

Q: What is Git?

A: Git is a distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Q: Why use Git?

A: Git is used for its efficiency, strong support for non-linear development (thousands of parallel branches), and its robust ability to handle large projects. It also supports distributed workflows, meaning each developer has a full history of the project.

Q: What are some key features of Git?

A: Key features include distributed development, strong support for branching and merging, data integrity, and the ability to handle large projects efficiently.

Installing Git on your local machine and setting up Git configurations

Q: How do you install Git on a local machine?

A: On Windows, you can use the Git installer from git-scm.com. On macOS, you can use Homebrew with the command `brew install git`. On Linux, you can use your distribution's package manager, e.g., `sudo apt-get install git` for Debian-based systems.

Q: How do you configure Git with your name and email?

A: Use the following commands:

`git config --global user.name "Your Name"` `git config --global user.email "your.email@example.com"`

Basic Git commands

Q: What does the `git init` command do?

A: `git init` initializes a new Git repository in the current directory.

Q: What does the `git add` command do?

A: `git add` adds files to the staging area, which prepares them to be included in the next commit.

Q: What does the `git commit` command do?

A: `git commit` takes the files in the staging area and permanently stores them in the repository's history.

Q: What does the `git status` command do?

A: `git status` shows the current status of the working directory and the staging area, including which changes are staged, unstaged, and untracked.

Q: What does the `git log` command do?

A: `git log` displays the commit history of the repository.

Creating a new Git repository on a local machine

Q: How do you create a new Git repository in a directory?

A: Navigate to the directory and run `git init`.

Initializing a Git repository in an existing project

Q: How do you initialize a Git repository in an existing project directory?

A: Navigate to the project directory and run `git init`.

Adding files to the staging area and committing changes

Q: How do you add a specific file to the staging area?

A: Use `git add <filename>`.

Q: How do you commit changes with a message?

A: Use `git commit -m "Your commit message"`.

Understanding branches in Git

Q: What are branches in Git?

A: Branches in Git allow you to create separate lines of development within a repository. They are used to develop features, fix bugs, or experiment in isolation from the main codebase.

Q: Why use branches in Git?

A: Branches enable parallel development and help manage workflows like feature development, bug fixes, and releases independently.

Creating and switching between branches

Q: How do you create a new branch?

A: Use `git branch <branch-name>`.

Q: How do you switch to an existing branch?

A: Use `git checkout <branch-name>`.

Merging branches and resolving conflicts

Q: How do you merge another branch into your current branch?

A: Use `git merge <branch-name>`.

Q: What should you do if there are conflicts during a merge?

A: Manually resolve the conflicts in the affected files, then use `git add <filename>` to mark them as resolved, and finally commit the changes.

Collaborating with others

Q: How do you pull changes from a remote repository?

A: Use `git pull`.

Q: How do you push changes to a remote repository?

A: Use `git push`.

Forking and submitting pull requests on platforms like GitHub or GitLab

Q: What does it mean to fork a repository?

A: Forking a repository means creating a personal copy of someone else's project on GitHub or GitLab, which allows you to freely experiment with changes without affecting the original project.

Q: How do you submit a pull request?

A: After making changes to your forked repository, navigate to the original repository on GitHub or GitLab and create a pull request from your fork's branch to the original repository's branch.

Exploring advanced Git topics

Q: How do you amend the last commit?

A: Use `git commit --amend`.

Q: How do you perform an interactive rebase?

A: Use `git rebase -i <base-commit>`.

Q: How do you create a Git alias?

A: Use `git config --global alias.<alias-name> '<git-command>'`.

Q: What is the difference between global and local Git configurations?

A: Global configurations apply to all repositories for a user and are stored in the `~/.gitconfig` file, while local configurations apply only to a specific repository and are stored in the repository's `.git/config` file. Use `git config --global` for global settings and `git config` for local settings.