GitTutorial

Runpeng Li

February 2024

Git Basics: A Practical Guide for Beginners

Git is a distributed version control system that helps you track changes to your codebase. By the end of this tutorial, your students will be well-equipped to use Git for their semester-long projects.

1. What is Git?

Git allows collaboration, keeps your project history organized, and facilitates seamless teamwork.

2. Installation and Configuration

- Install Git: Download it from the official website or use package managers like Homebrew (for macOS) or Chocolatey (for Windows).
- Configure Git: Set up your name and email using the following commands:

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

3. Creating a Repository

- Initialize a New Repository: Navigate to your project folder and run:

```
git init
```

- Cloning an Existing Repository: Use:

```
git clone <repository_url>
```

4. Basic Workflow

- Adding Changes: Add files to the staging area before committing them:

  ```
  git add <file1> <file2>
  ```

- Committing Changes: Commit your staged changes with a descriptive message:

  ```
  git commit -m "Add feature X"
  ```

- Pushing and Pulling: Push local commits to the remote repository:

  ```
  git push origin <branch_name>
  ```

  Pull changes from the remote repository:

  ```
  git pull origin <branch_name>
  ```

5. Branches and Merging

- Creating a Branch: Create a new branch for a feature or bug fix:

  ```
  git checkout -b feature/my-feature
  ```

- Switching Branches: Move between branches:

  ```
  git checkout <branch_name>
  ```

- Merging Branches: Merge changes from one branch into another:

  ```
  git merge <source_branch>
  ```

- Handling Conflicts: Resolve conflicts manually and commit the changes.

6. Stashing

- Stash Changes: Temporarily save uncommitted changes:

  ```
  git stash save "Work in progress"
  ```

- Apply Stash: Retrieve stashed changes:

  ```
  git stash apply
  ```

7. Additional Resources

- GitHub: Explore GitHub for collaborative development and hosting repositories.
- Practice: Create a sample project, experiment with branches, and practice merging.

Remember, Git is a powerful tool, and practice makes perfect. Happy coding!