Here’s a comprehensive list of **Git commands** commonly used for version control and collaborative development, along with explanations:

**1. Setup and Configuration**

* **Initialize a Git Repository**:

bash

Copy code

git init

Creates a new Git repository in the current directory.

* **Set User Information**:

bash

Copy code

git config --global user.name "Your Name"

git config --global user.email "your.email@example.com"

Configures your name and email for commits.

* **View Configuration**:

bash

Copy code

git config --list

Displays the current configuration settings.

**2. Working with Repositories**

* **Clone a Repository**:

bash

Copy code

git clone [repository URL]

Downloads a repository and its history to your local machine.

* **View Repository Status**:

bash

Copy code

git status

Shows the status of your working directory (e.g., changes, staged files).

* **List Remote Repositories**:

bash

Copy code

git remote -v

Lists the URLs of the remote repositories linked to your project.

**3. Staging and Committing Changes**

* **Stage Changes**:

bash

Copy code

git add [file]

git add . # Stages all changes in the current directory

Adds changes to the staging area.

* **Commit Changes**:

bash

Copy code

git commit -m "Commit message"

Records staged changes in the repository.

* **Amend the Last Commit**:

bash

Copy code

git commit --amend -m "Updated commit message"

Updates the last commit with new changes or a revised message.

**4. Branching**

* **Create a New Branch**:

bash

Copy code

git branch [branch-name]

Creates a new branch.

* **Switch to a Branch**:

bash

Copy code

git checkout [branch-name]

Moves to the specified branch.

* **Create and Switch to a New Branch**:

bash

Copy code

git checkout -b [branch-name]

Combines branch creation and switching.

* **List All Branches**:

bash

Copy code

git branch

Displays all branches in the repository.

* **Delete a Branch**:

bash

Copy code

git branch -d [branch-name]

Deletes the specified branch (if merged).

**5. Merging and Rebasing**

* **Merge a Branch into the Current Branch**:

bash

Copy code

git merge [branch-name]

Integrates changes from another branch into the current branch.

* **Rebase the Current Branch**:

bash

Copy code

git rebase [branch-name]

Reapplies commits on top of another base commit, creating a linear history.

**6. Collaboration**

* **Pull Changes from a Remote Repository**:

bash

Copy code

git pull [remote-name] [branch-name]

Fetches and integrates changes from a remote repository.

* **Push Changes to a Remote Repository**:

bash

Copy code

git push [remote-name] [branch-name]

Uploads local changes to a remote repository.

* **Add a Remote Repository**:

bash

Copy code

git remote add [remote-name] [repository URL]

Links a remote repository to your project.

* **Fetch Changes from Remote**:

bash

Copy code

git fetch [remote-name]

Retrieves changes from a remote repository without merging.

**7. Undoing Changes**

* **Discard Local Changes**:

bash

Copy code

git checkout -- [file]

Restores a file to the last committed state.

* **Unstage a File**:

bash

Copy code

git reset [file]

Removes a file from the staging area.

* **Revert a Commit**:

bash

Copy code

git revert [commit-hash]

Creates a new commit that undoes the changes of a specified commit.

* **Reset to a Previous Commit**:

bash

Copy code

git reset --soft [commit-hash] # Keeps changes staged

git reset --mixed [commit-hash] # Keeps changes in the working directory

git reset --hard [commit-hash] # Deletes all changes

**8. Viewing and Inspecting**

* **View Commit History**:

bash

Copy code

git log

git log --oneline # Compact view

* **Show Changes in Files**:

bash

Copy code

git diff

git diff [file]

* **Show a Specific Commit**:

bash

Copy code

git show [commit-hash]

**9. Tags**

* **Create a Tag**:

bash

Copy code

git tag [tag-name]

* **Push Tags to Remote**:

bash

Copy code

git push [remote-name] [tag-name]

* **List Tags**:

bash

Copy code

git tag

**10. Advanced Commands**

* **Stash Changes**:

bash

Copy code

git stash

Temporarily saves changes without committing.

* **Apply Stashed Changes**:

bash

Copy code

git stash apply

* **Resolve Merge Conflicts**: After a merge conflict occurs:

bash

Copy code

git status # Shows conflicting files

# Edit files to resolve conflicts

git add [file] # Mark conflicts as resolved

git commit

These commands cover most scenarios for managing code and collaborating effectively using Git. Let me know if you need further clarification or help with specific workflows!

4o

Top of Form

Bottom of Form