Title: Git User Guide

Table of Contents

1. Introduction

Git is a distributed version control system designed to handle everything from small to very large projects with speed and efficiency. This guide covers basic usage, common issues, and troubleshooting steps.

---

2. Getting Started with Git

Installation

- **Windows**: Download from Git for Windows and follow the installer instructions.
- **macOS**: Install via Homebrew using `brew install git`.
- **Linux**: Install using the package manager, e.g., `sudo apt-get install git` for Debian-based distributions.

Basic Configuration

Set up your Git environment with your name and email:

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

3. Common Git Commands

Cloning a Repository

To clone a repository:

```
git clone https://github.com/username/repository.git
```

Committing Changes

Stage files for commit:

```
git add filename
```

1. Commit changes:
   ```
   git commit -m "Commit message"
   ```
2. Pushing Changes:

   To push your commits to a remote repository:

   ```
   git push origin branch-name
   ```

   Pulling Changes

   To fetch and merge changes from the remote repository:

   ```
   git pull origin branch-name
   ```

4. Common Issues and Troubleshooting

Merge Conflicts

If you encounter merge conflicts, Git will mark the conflicts in the files. Resolve the conflicts manually and then:

```
git add resolved-file
git commit
```

Detached HEAD State

If you find yourself in a detached HEAD state, switch back to a branch:

```
git checkout branch-name
```

Authentication Issues

If you face authentication issues, ensure your credentials are correct. For HTTPS, you may need to use a personal access token.

5. Advanced Topics

Branching and Merging

To create a new branch:

```
git branch new-branch
```

To switch branches:

```
git checkout new-branch
```

To merge a branch into the current branch:

```
git merge branch-name
```

Rebasing

To rebase changes from one branch onto another:

```
git rebase branch-name
```

Cherry-Picking

To apply a commit from one branch to another:

```
git cherry-pick commit-id
```

6. Best Practices

Commit Messages

Use clear and descriptive commit messages. For example:

sql

Copy code

```
Add user authentication feature
```

Branch Naming Conventions

Use descriptive names for branches, such as:

- `feature/added-login`
- `bugfix/fixed-crash`

7. Appendix

Git Glossary

- **Repository**: A directory that contains your project files and the entire history of changes.
- **Commit**: A snapshot of your files at a specific point in time.

Additional Resources

- Official Git Documentation
- Pro Git Book