GitHub Tutorial

Wilson Schwegler
Schwegler@oxy.edu
Occidental College

1 Introduction

GitHub is a powerful software development platform. It is used for storing, tracking, and collaborating on projects. GitHub makes it easy for developers to work seamlessly on projects of any scale with fellow developers. In this tutorial, we will explore the fundamentals of GitHub using command line.

2 Cloning

Cloning is the process of creating a local copy of a remote repository from GitHub. When a repository is cloned, the entire project is downloaded, including its files, commit history, and branches to your local machine. This enables you to work on the project, make changes, and contribute without altering the original repository.

2.1 How To

The command for cloning a repository from GitHub using command line is:

```
git clone repository_URL
```

The repository URL can be found on the GitHub website. Navigate to the repository that is to be cloned. Once on the repository page, look for the green ”Code” button. Click on the button to reveal the drop-down menu that will contain the URL under ”HTTPS”. The URL will start with ”https://github.com/”. After successfully cloning the repository, you will have a local copy of the entire project on your machine.

3 Adding

Adding is the process of selecting and staging specific changes made to your local repository. Developers use adding to include changes or modifications that they made to files in the staging area. This allows developers to pick and choose which modifications they would like to include in the upcoming commit (committing will be covered in the next section) to the remote repository. Developers have total control over what contents are included in their commits to ensure that only the intended modifications are recorded in the project’s history and remote repository.

3.1 How To

There are two commands that can be used in command line for adding. The first is:

```
git add file(s)
```

This command is used to stage-specific files by replacing file(s) with the name of the file or files you would like to add. If you would like to add all changes in the working directory, you can use the command:

```
git add .
```

After your desired files are added then they are staged, meaning they are marked to be included in the next commit. The staging area is a pre-commit area where you can review and fine-tune the changes you want to include in the next commit. At this point, the changed files are not yet recorded in the version history but simply prepared for the next commit.

4 Committing

Committing is the process of saving what you have staged in the local repository. Once the modifications have been added to the staging area, committing is employed to record these changes in the version history. A commit is accompanied by a descriptive commit message that explains the purpose of the changes. Committing is crucial when collaborating with other developers as it creates a well-documented and organized project history that allows others to track the evolution of the project and provides context about the changes introduced in the commit.

4.1 How To

The command for committing to a GitHub repository using command line is:

```
git commit -m "Your commit message here"
```

Replace ”Your commit message here” with an informative message describing the purpose of the commit and other important information.

5 Pushing

Pushing is the process of uploading the committed changes from the local repository to the remote repository hosted on GitHub. Pushing is crucial as it allows fellow developers to share their contributions in a centralized repository to ensure the latest modifications are reflected in the shared project.

5.1 How To

The basic command for pushing to a GitHub repository using command line is:

```
git push
```

This command will upload your committed changes to the remote repository. If you are working on a specific branch (discussed later on) you will use the command:

```
git push origin Branch Name
```

Replace Branch Name with the name of the branch you want to push to.

6 Pulling

Pulling is the process of fetching and merging changes from the remote repository to the local repository. Pulling is essential for keeping your local repository up-to-date with the latest modifications made by other developers.

6.1 How To

The basic command for pulling from a GitHub repository using command line is:

```
git pull
```

This command will fetch the changes from the remote repository and merge them to the local copy. If pulling from a specific branch, use the command:

```
git pull origin Branch Name
```

Replace Branch Name with the name of the branch you want to pull from.

7 Stashes

Stashing is a feature that allows developers to temporarily save changes in their working directory without committing them. This enables developers to stash modifications that are not ready to be committed and switch to a different branch without committing incomplete work.

8 Branches

Branches are a version of the repository that is an independent line of development. This allows multiple developers to work on different aspects of a project simultaneously. Branching facilitates parallel development, allowing teams of developers to collaborate and make changes without affecting the main project. Once a branch is ready, it can be merged into the main branch.

9 Merging

Merging is the process of combining changes from one branch into another. When developers finish their work on a branch they can merge their branch into the main branch. This allows for consolidation of different lines of development and ensures that the latest finished code is incorporated into the main project. GitHub makes merging easy and efficient, however conflicts may arise.

9.1 Dealing With Conflicts

When multiple developers make changes that are conflicting to the same file, a merge conflict will arise. GitHub will mark these conflicts and developers will need to choose which changes they will keep. After developers resolve these conflicts and commit the changes, the project is ready for continued development.