STA303 - Github Learning and Repository

Shiqi Li

Github Learning Notes

Git and Github

Git is a version control system / technology to keep track of source code history, and **GitHub** is a public platform that supports / realizes this technology.

The `.git` and `.gitignore`

The `.git` is the suffix for a SSH connection to remote git repo, represents the whole repository.

The `.gitignore` file is a text file where each line contains a pattern for files / directories to ignore, generally placed at the root folder of the repository.

Remote and local repository

Remote Git repository is the one that is stored in a non-local address (in the internet), while local git directory is the local directory that is connected to the remote repo.

Working Directory, Staging Area, and Repository

Working Directory contains the files currently working on, staging area contains files tracked by git (git status), repository is basically every file on the `.git` directory.

Command Line Tools

1. Install and check version of `Git` on a Mac:

```
brew install git
git --version
```

2. Clone a remote repository into local address:

```
git clone [repo_address_url]
```

2. To add local files to the working directory:

```
git add [file_name]
```

3. Commit file changes from the staging area to the repository

```
git commit -m "message"
```

4. Push a local repo to a remote repository

```
git remote add origin -address-
git push
```

Branching

Besides version control using development timeline, Git also provides a horizontal dimension to keep track of different running version of the application. Usually, there are three types. A **master** branch is holding the current version of the complete running application that is ready for use; a **develop** branch that incorporates the features to be included in the next version and will replace the master branch for the next version release; and multiple **feature** branches that each represents a new feature being added to the application. Sometimes multiple feature branches will be first merged into a **release** branch and then update to the develop branch after reviews and bug fixes.

Usually, when a new feature implementation is finished, the developer will push the code to the feature branch, and submit a pull request, and after the senior people / code reviewer / other team member finishes the code review, this feature branch will be merged into the develop branch.

This standardized git workflow that defines a branching model for project implementation, usually composed of master, develop, release, feature is called **GitFlow**.

Some command line tools for Git branching.

1. Create a new branch:

```
git checkout -b *branch_name*
git push --set-upstream origin develop
git push
```

2. Switch to a difference branch

```
git checkout *new_branch*
```

Difference between `git checkout` and `git branch`: `git checkout` (without `-b`) is to switch to a branch while `git branch` is to create a new branch.

Interacting with GitHub

There are three ways to interact (update, set up, or modify a repository) with the GitHub remote repository:

1. Go to the GitHub website and log in to personal profile, then simply delete or add files.
2. Using command line tools to set up an upstream connection between local repo and remote.
3. Generate a SSH key on local machine and then copy it to GitHub for encrypted connection.

My GitHub Repository

Following the standard nested structure of a directory, I have re-arranged my STA303 files and created my own GitHub repository. This repository is also planned to be used to contain future coursework projects, so that I can have a personal URL to showcase my projects and skills to potential employers. I did not include any mixed-assessment materials to avoid potential academic integrity problems.

I have also supplied the README.md file in the root directory. My GitHub repository can be publicly accessed at: