Resources: Using git

Just Enough git

Terminal Commands for git

Configure Your Computer Account to be Trusted (One-Time Operation)

Open a terminal. Try this command:

```
cat ~/.ssh/id_rsa.pub
```

If you get a message that says `No such file or directory`, then you need to create an SSH key for your account. Only do this if the command above failed.

```
ssh-keygen
```

This should create a public/private key pair for you to use in secure SSH communications. Go back and run the `cat` command from above. This should give a few lines of text. Copy this text into your clipboard.

If you don’t have one, create an account at the site where your project will reside, such as GitHub. Go to your account settings (top-right click the account and menu will appear, select “settings”). Select “SSH and GPG keys”. Select “New SSH key”, and paste the key from your clipboard. Give a descriptive name to the key so you know where it came from.

You only need to repeat this process if you want to use git on another computer or with a different user on the same computer.

Clone the Repository (One-Time Operation)

First, you need a URL for the project you want to work with. You can create a new project at sites such as GitHub. Or, you may be working with an existing repository set up as part of your course. If you visit the website for the project, find the “Clone or download” button and click it. Be sure you have selected the “Use SSH” option, and copy the URL to you clipboard.

In the terminal, first `cd` to the directory you want to store the code in. Then clone the respository. For example

```
mkdir ~/projects
cd ~/projects
git clone THE_URL_OF_THE_REPOSITORY
```

This will create a complete copy of the project on your computer. You should not need to repeat this step unless you have another repository you want to work with.

Edit Files

You can now edit the content of the repository by freely creating files and directories, deleting files and directories, and editing the content of files.

Commit Changes

After you have made a set of changes that you would like to keep, you need to stage the changes, and commit them to the repository. The quickest way to stage the changes is to add all changes you have made (whether adding, editing or deleting) with one command. Do this from the top directory of your repository.

```
 git add -A .
```

Note that more fine-grained options exist for staging only specific changes. You should learn to use them soon. But, this will get you started.

Next, you need to make the staged changes permanent by committing them to the repository. You should give a brief message describing the purpose of the changes that are being committed when you commit.

```
git commit -m 'YOUR MESSAGE HERE'
```

Your changes are now part of the history of this project. Sometime soon, you'll want to learn how to use `git log` to review a project's history.

Synchronize the Local and Remote Repositories

If work on this project from multiple computers, or if there are multiple people working on the project, you'll want to synchronize changes from the different sources. Even if you only work on one computer, you'll want a backup of your project kept at the remote site.

To fetch changes from the remote repository into your local computer, pull them from the server.

```
git pull -u origin
```

To send changes to the remote repository, push them to the server.

```
git push -u origin
```

If there are changes in the remote repository that haven't been pulled into your local repository, you'll have to pull them before you are allowed to push your changes.

Sometimes there are conflicts when you pull changes from the remote repository. Most of the time, git can resolve these conflicts automatically. If it can't you'll have to fix the problem. Find help the first time you have to merge conflicts.