Lab Activity 2

Repositories and Project Setup

Objective

You will get familiar with git and GitHub individually and setup your team repository.

Overview

As you work on your team project you will have to share code with your teammates and combine your code into a single functioning project. There are many ways to share this code (email, Google Docs, flash drives, etc) but we will explore an effective way to collaborate on a project by using version control software called git and a hosting service called GitHub. By using git and GitHub you will be able to share and combine code effectively.

We'll use the following two commands to download code from a repository:

1. **Clone** a repository: Download a complete copy of the repository onto your laptop
2. **Pull** commits: To download the latest changes made by your teammates you should pull their commits using the blue arrow, or clicking VCS -> git -> pull. You should pull often when working on your project to ensure your code is compatible with your teammates contributions

To share code using git we'll go through 3 steps, though IntelliJ simplifies this process:

1. **Add** files: This tells git that it should be tracking changes made to the files that are added. IntelliJ typically asks if you want add a file to git whenever you create a new file
2. **Commit** changes: This tells git that it should sync any changes made to all added files in your copy of the repository on your laptop. Whenever you commit you will be asked to write a commit message about the changes you've made
3. **Push** commits: To share your changes with your team you will push your commits to GitHub. Any commits that are not pushed cannot be seen by your team so be sure to push when you commit. IntelliJ has a commit and push option which can be used to avoid a click. If you commit without pushing go to VCS -> git -> push to push

Part 1: Retrieving Code From a Repository (25 Points)

You may work with you team on this part of the lab, but **all** team members must complete this part on their laptop and verify it with a TA.

I have created a git repository containing many of the Scala examples you've seen in lecture and shared it on GitHub. For this part of the lab you will use git to download my code and run it on your laptop.

Repository URL: https://github.com/hartloff/CSE116-Scala-Examples.git

1. In IntelliJ clone this repository and create a new project with this code
    a. https://www.jetbrains.com/help/idea/manage-projects-hosted-on-github.html
2. Run Maven in the new project
    a. I am using a library that we have not seen yet. Be sure to use Maven to download it just as you did with scalatest
3. Run the object Activity2 in the lab package
    a. Once your entire team has this program running show a TA and move on to part 2

Troubleshooting:

- If the src folder is not blue you may have to right-click it -> Mark Directory As -> Sources Root.
- If you're running Windows you'll have to install git and setup your email/username

*You'll see a blue arrow in the menu bar in IntelliJ. Clicking this will download the latest changes made to this repository. Use this button to get the latest lecture example on your laptop

**There is also a python repository https://github.com/hartloff/CSE116-Python-Examples.git

Part 2: Sharing a Team Repository (25 Points)

Now that you know how to download a git repository it's time to create and share your own. In this part you will create the repository that you will use for your team project and each member will push content into this repository.

1. Create a GitHub account if you don't have one already (It's free)
2. Have only 1 team member setup and share the repository
    a. Create new project in IntelliJ
    b. Follow these steps to share that project on GitHub (IntelliJ will ask for your GitHub username/password then use the GitHub Web API to share your project)
3. All other team members will share content with the repository
    a. Clone the repository just as you did in part 1
    b. Make any edit to the project
    c. Find the green check mark in the menu and click this to share your code
        i. Write a meaningful commit message when asked
        ii. Select commit and push to share your changes with your team
        iii. https://www.jetbrains.com/help/idea/commit-and-push-changes.html
4. When you can see all team members changes in GitHub you have completed this part

a. Verify with your TA that you are finished
b. **On a sheet of paper write the names of each member of your team who completed both parts of the activity and hand it in to a TA to earn credit for the activity**

Overview: To share code using git we went through 3 steps, though IntelliJ simplifies the process

4. **Add** files: This tells git that it should be tracking changes made to the files that are added. IntelliJ typically asks if you want add a file to git whenever you create a new file
5. **Commit** changes: This tells git that it should sync any changes made to all added files in your copy of the repository on your laptop. Whenever you commit you will be asked to write a commit message about the changes you've made
6. **Push** commits: To share your changes with your team you will push your commits to GitHub. Any commits that are not pushed cannot be seen by your team so be sure to push when you commit. IntelliJ has a commit and push option which can be used to avoid a click. If you commit without pushing go to VCS -> git -> push to push

3. **Pull** commits: To download the latest changes made by your teammates you should pull their commits using the blue arrow, or clicking VCS -> git -> pull. You should pull often when working on your project to ensure your code is compatible with your teammates contributions