

Lab 1- Introduction to Git and Java Programming

Introduction

The purpose of this lab is to introduce you to using Git and two alternative ways of creating and executing Java programs.

Part I – Git and GitLab

If you forget the general working pipeline of using Git, please refer to the *slide-2* and *Using Eclipse along Git for Lab Assignments* in Week 1 Optional Reading. Slide-2 and Using Eclipse along Git for Lab Assignments can be found on Canvas.

Git is a version control protocol that allows you to manage and track changes to your programming projects over time. There are many advantages of using Git to manage your software development. The most important two include the easiness of reverting to a previous version of your project and coordination of collaborations. For how to use Git, please read support.cs.wvu.edu/index.php/Git. Please note that you may need to use command line tools, such as ssh to synchronize your local project to the mirror project on GitLab. If you have little experience using command line tools, please read support.cs.wvu.edu/index.php/CS_Technical_Survival_Guide first. This lab section serves as a stepwise guidance of how to use Git and Gitlab, and paves for the future labs and projects.

First, register an account on gitlab.com, confirm your registration, and sign into your account.

GitLab.com

GitLab.com offers free unlimited (private) repositories and unlimited collaborators.

- [Explore projects on GitLab.com](#) (no login needed)
- [More information about GitLab.com](#)
- [GitLab.com Support Forum](#)

By signing up for and by signing in to this service you accept our:

- [Privacy policy](#)
- [GitLab.com Terms](#).

Sign in	Register
Full name	
<input type="text"/>	
Username	
<input type="text"/>	
Email	
<input type="text"/>	
Email confirmation	
<input type="text"/>	
Password	
<input type="text"/>	
Minimum length is 8 characters	

Second, create a project. Name it with your name in the format as “first name-last name”. For instance, if your name is Rick Sanchez, the project should be named as “rick-sanchez”. Please note that you are required to keep the visibility level as “Private”.

Create from template [?](#)

Blank Ruby on Rails Spring NodeJS Express

OR

Import project from

GitHub Bitbucket Google Code Fogbugz

Gitea git Repo by URL GitLab export

Project path: `https://gitlab.com/` `qianghao`

Project name: `my-awesome-project`

Want to house several dependent projects under the same namespace? [Create a group](#)

Project description (optional)

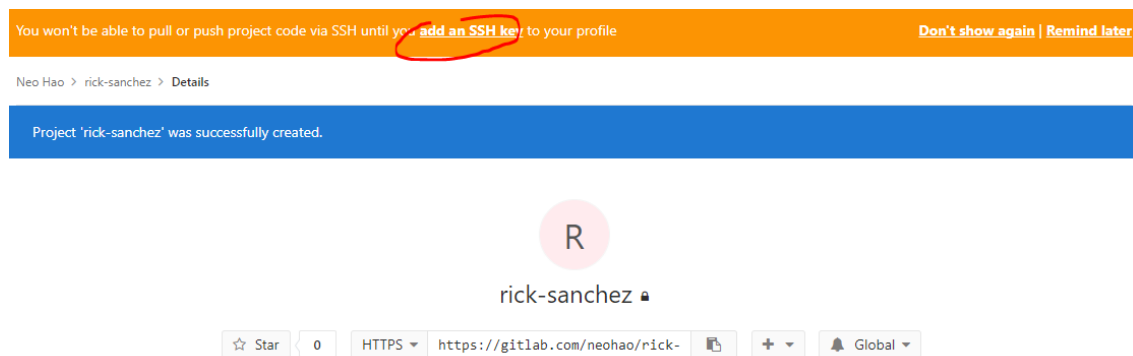
Description format

Visibility Level [?](#)

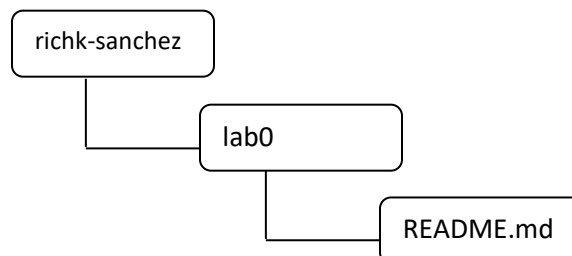
☒ Public ☒ Private

Project access must be granted explicitly to each user.

After you create the private repository as instructed, you will note a yellow bar notifying you to add an SSH key to your profile. Click the link and follow the instruction on the linked webpage. You will be able to use SSH to push / pull codes to your project. If you did not use SSH before, please refer to support.cs.wvu.edu/index.php/Getting_Around_the_Command_Line#SSH.

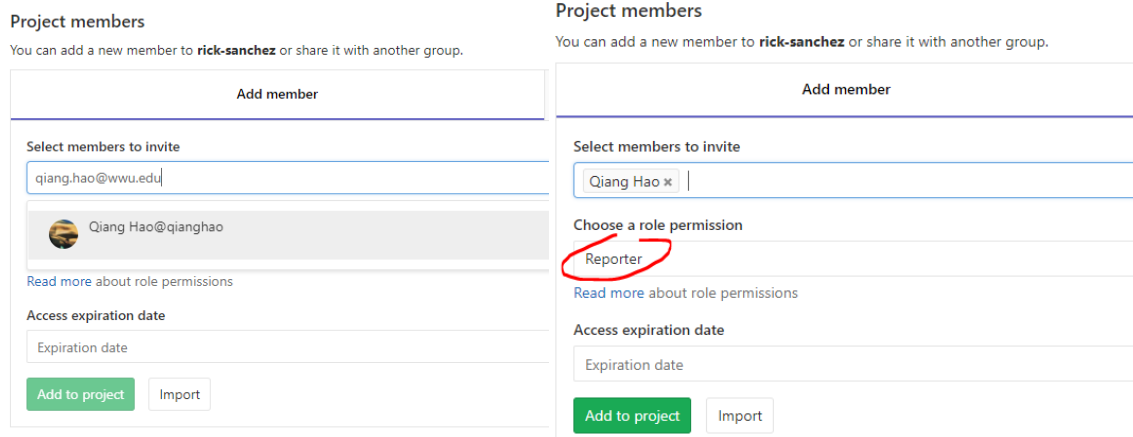


Third, clone the empty project you created on GitLab to your local drive. If you don't know how to do so, please read support.cs.wvu.edu/index.php/Git. In the cloned local folder, create a subfolder named "lab0", and add a file named "README.md" in lab1 in the subfolder. Put the following texts into README.md: *This is a description of lab0*. md file can be edited by any text editors.



To test the knowledge you learnt from reading support.cs.wvu.edu/index.php/Git, please commit the changes to your local project and push the changes to the mirror repo on GitLab.

Fourth, add the instructor (qiang.hao@wwu.edu) and the TA (tranw@wwu.edu) as reporters to your project.



Part II & Part III

All activities and files of sections Part II and Part III need to go under a **subfolder** named “**lab1**” in your GitLab project folder.

Part II - Using Eclipse to create your first Java program

Eclipse is an excellent IDE for Java programming. Using Eclipse to code Java can make life much easier for you, especially if you never coded in Java before. Features of Eclipses, such as auto correction, auto completion, and syntax error detection can allow you to focus more on mastering the essential skills of coding.

To use Eclipse along Git, you need to set the working directory of Eclipse as the cloned local project folder. For detailed guidance, please refer to the *slide-2* and *Using Eclipse along Git for Lab Assignments* in Week 1 Optional Reading on Canvas. After that, you can create different Eclipse project for each lab/group project within Eclipse, and code only in Eclipse. The command line tools can be only used for synchronization between your local folder and its mirror on GitLab.

You are required to create your first Java program that prints “Hello, world!” using Eclipse.

The HelloWorld program source code is provided for you:

```
public class HelloWorld {
    public static void main(String [] args) {
        System.out.println("Hello, world!");
    }
}
```

Part III – Java Programming with a Plain Text Editor and Terminal

Another way to write Java source code is by using a plain text editor instead of an IDE such as *Eclipse*. In this part of the lab, you will create another Java program that prints “Hello, world!” using Command Line Tools, such as the Terminal on Linux / Unix machine.

The HelloAgain program source code is provided for you:

```
public class HelloAgain {  
    public static void main(String [] args) {  
        System.out.println("Hello Again, world!");  
    }  
}
```

Part IV – Research Time

Here are the codes that print Hello, World in Java:

```
public class Hello {  
    public static void main(String [] args) {  
        System.out.println("Hello, world!");  
    }  
}
```

Explain key works in the hello world java program by answering the following questions. You are supposed to do some research on these questions by reading the manual of Java or searching online. The answers to the 6 questions should not go beyond 1 page.

1. What does a semicolon indicate in Java program?
2. What does main method do? When do you want a class to have main method?
3. How do static fields and methods work? Why do you need static fields or methods?
4. What access type can a method have in addition to "public"? When do you use each type?
5. What is "String[] args"? What does it do?
6. Why does it take "System.out.println" to print a line? Explain the relationship among System, out and println.

Submission and Grading

It is required that you organize your local labs in a single folder, and synchronize the folder to a repository on GitLab. It is also required that you keep synchronizing the two by periodical committing/pushing your local changes to GitLab. **30 points would be deducted if there are no commit/push in your GitLab repository by Oct. 5th.**

Please use comments to explain your program or any sections that are possibly difficult to understand. **10 points would be deducted if few or no comments can be found in your java files.**

The lab will be graded according to the following guidelines.

- A score between 0 and 100 will be assigned.
- If the source file(s) are not submitted by the deadline, then a grade of 0 will be assigned.
- 10 points will be deducted for late labs per day.
 - 0-24 hours late -10
 - 24-48 hours late -20
 - >48 hours late = 0

After you have completed and thoroughly tested your program, upload a **.zip** file containing the three required files to **Canvas**:

- **HelloWorld.java**
- **HelloAgain.java.**
- **researchTime.pdf**