

COMP3111: Software Engineering

Introduction to Git and GitHub

Learning Outcomes

- Be familiar with the steps of creating a local repository of a Java project via Eclipse
- Be able to learn how to commit changes and other Git operations
- Be able to create a GitHub account and track changes in the remote repository

Software Development Environment:

- **Java SE Development Kit 15 (JDK 15.0.2):** You may need an Oracle account.
<https://www.oracle.com/java/technologies/javase/jdk15-archive-downloads.html>
- **Eclipse IDE for Java Developers 2020-12 R**
<https://www.eclipse.org/downloads/packages/release/2020-12/r>
- **GitHub** (<https://www.github.com>)

Make sure that you have downloaded and installed the recommended versions for both JDK (15.0.2) and Eclipse (2020-12 R) on your machine. We have identified some incompatibilities with the latest versions (JDK + Eclipse) on Mac machines, so we suggest using the combination of these older versions for the work on the lab tasks and project.

Guided Lab Exercises

There are four guided exercises in this lab:

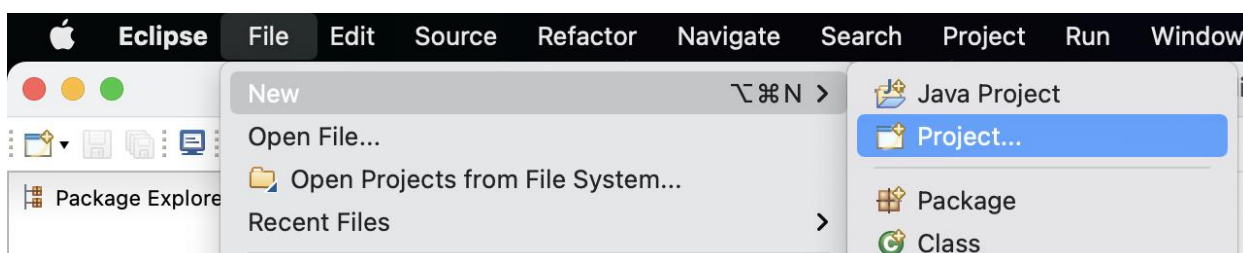
- **Exercise 1: Create a Gradle Java Application project in Eclipse;**
- **Exercise 2: Create a local Git repository;**
- **Exercise 3: Commit to your local Git repository;**
- **Exercise 4: Setup a remote Git repository on GitHub and push to it.**

To earn the credit points for this lab, you must complete the lab assignment and post the correct answers (as two personalized URLs) through Canvas by the due date. You should first go through all the guided exercises to better prepare for the lab assignment.

During the lab session, the TA will go through the guided exercises and address any issues and questions regarding the guided exercises and the lab assignment. You can demonstrate your work and ask the TA to verify whether you have got the correct answer (or pointing out the errors). Please note that even if you might have demonstrated your work to the TA during the lab session, you still need to make an online submission (as two personalized URLs) through Canvas.

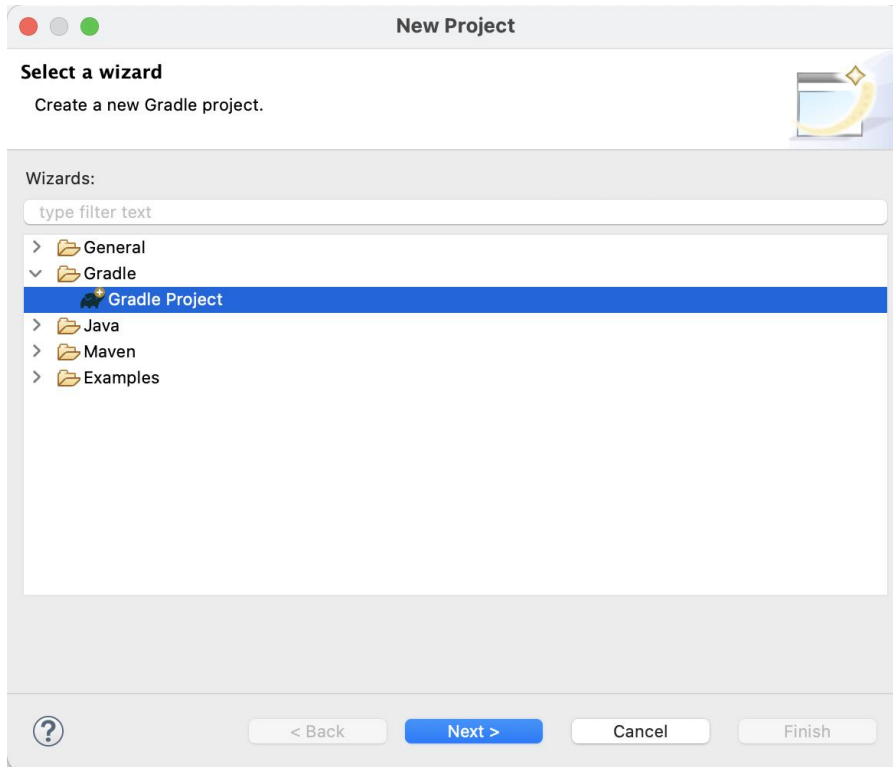
Exercise 1: Create a Java project in Eclipse

Step 1.1: From the menu bar, select Project

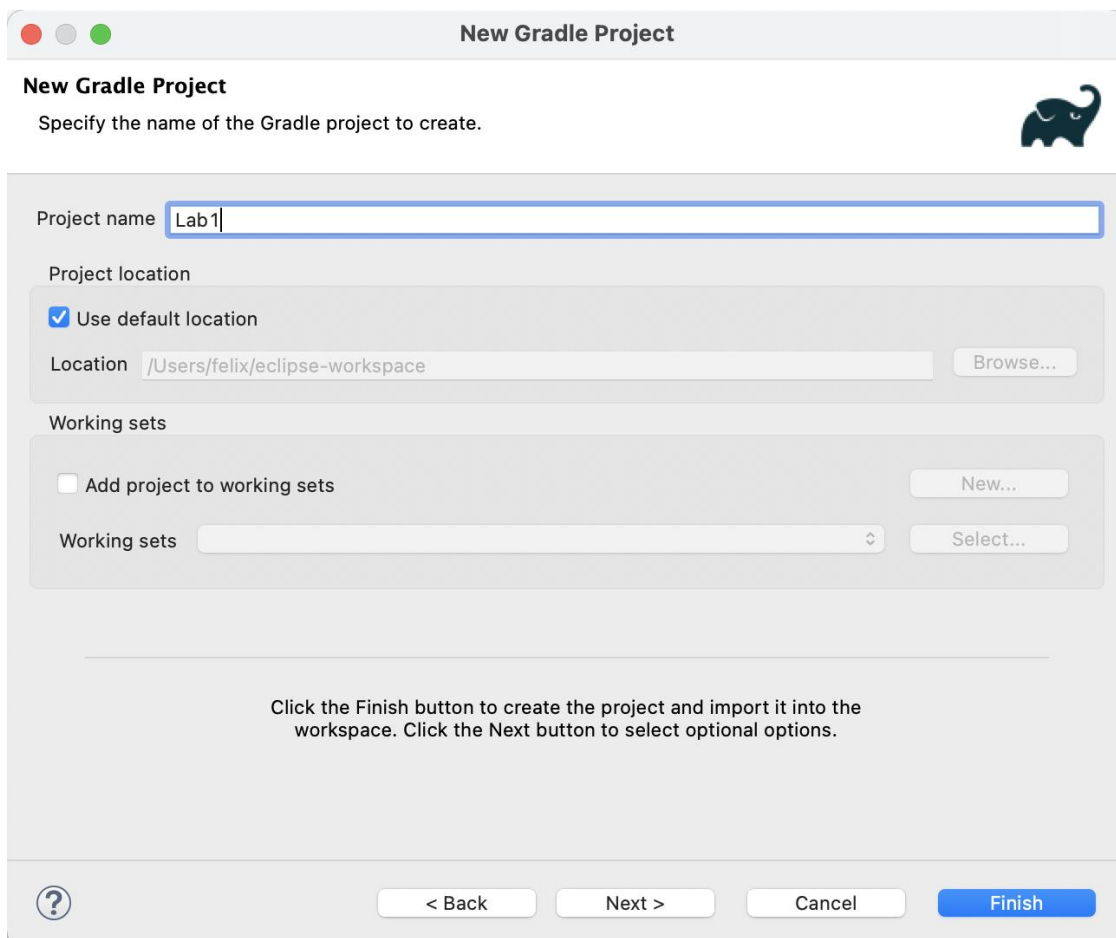


Step 1.2: Click Gradle > Gradle Project

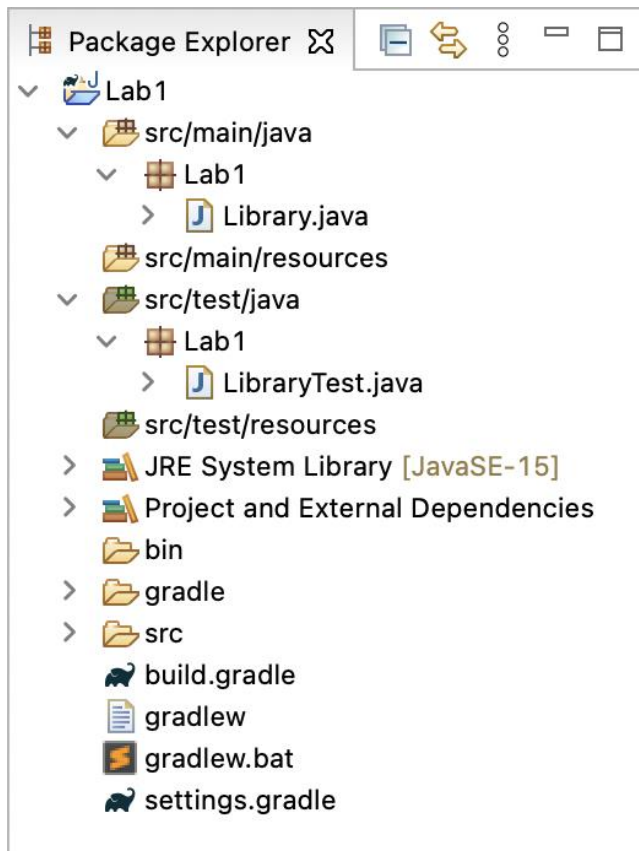
Note: Gradle works like Makefile in C++ so that we can streamline a lot of tasks using Gradle



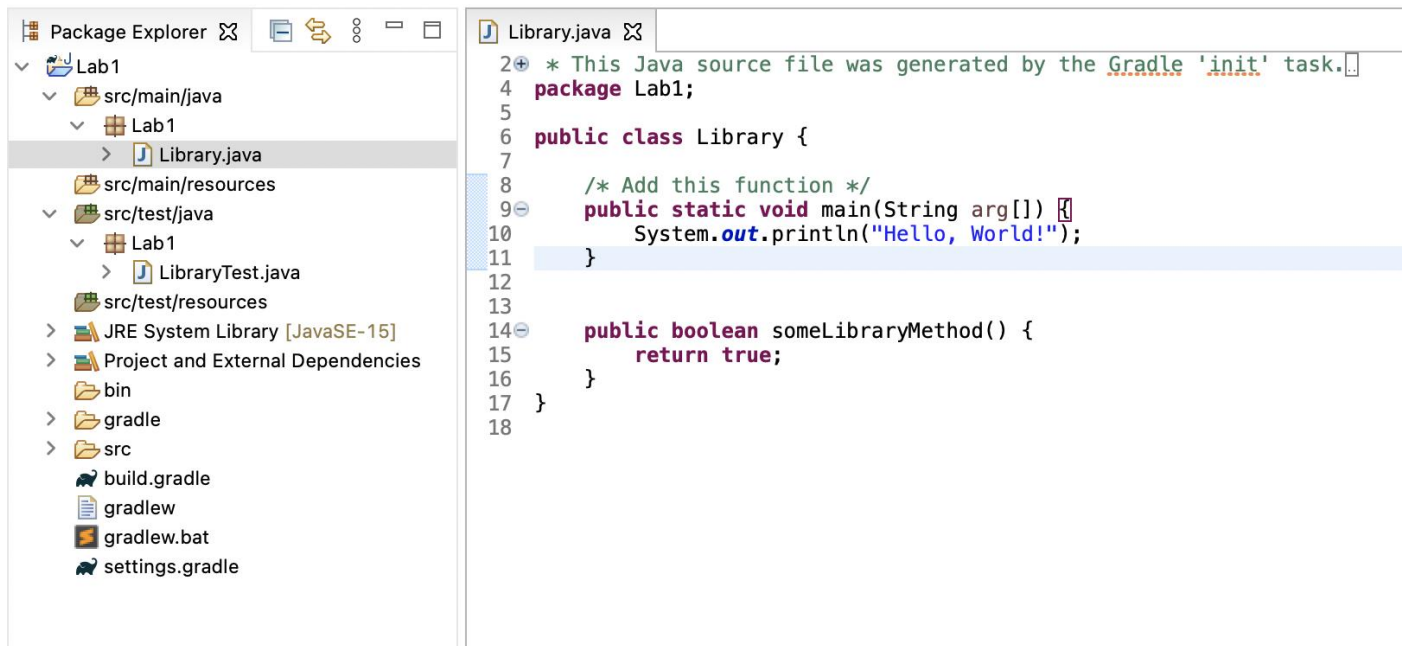
Step 1.3: Type “Lab1” for the project name and click “Finish”



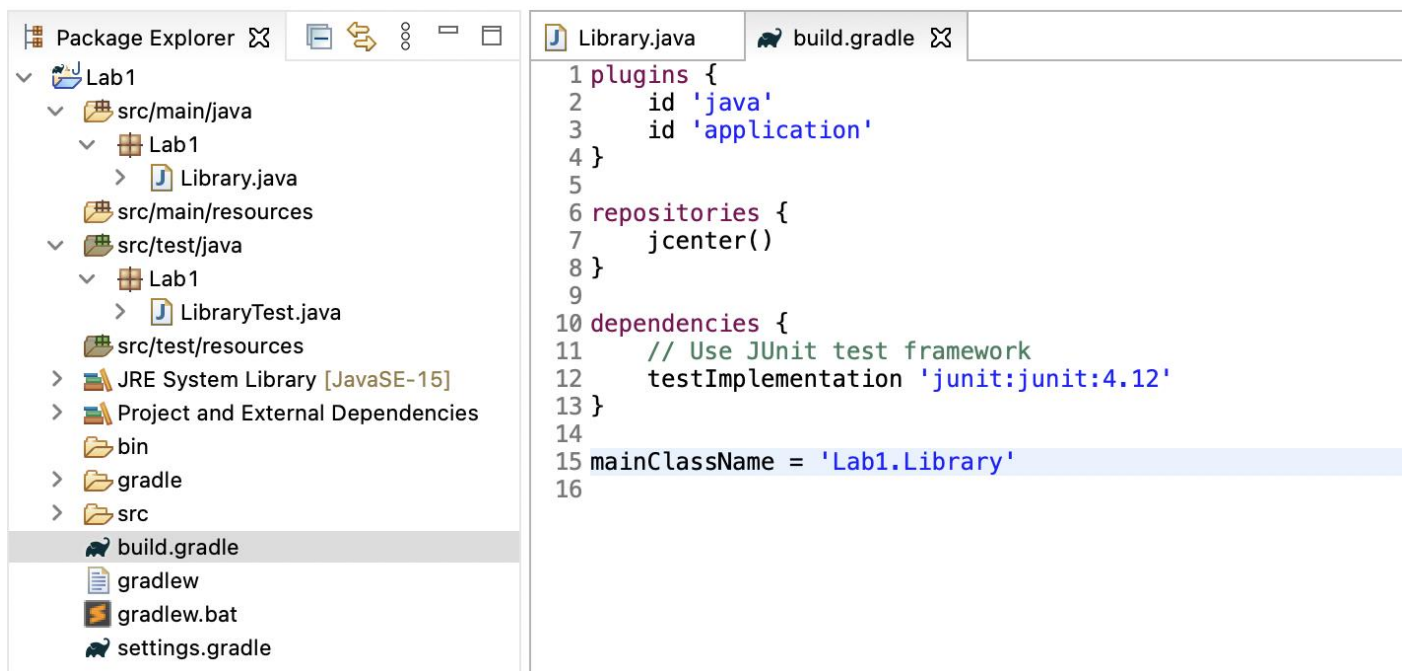
Step 1.4: A new Gradle project should be created for you. It will generate two java files for you. They are located at `src/main/java` and `src/test/java`. As the name suggests, the files stored in the `src/test/java` folder are for testing purpose. At the moment, we should ignore this folder and the files inside. Compare your project against the following screenshot.



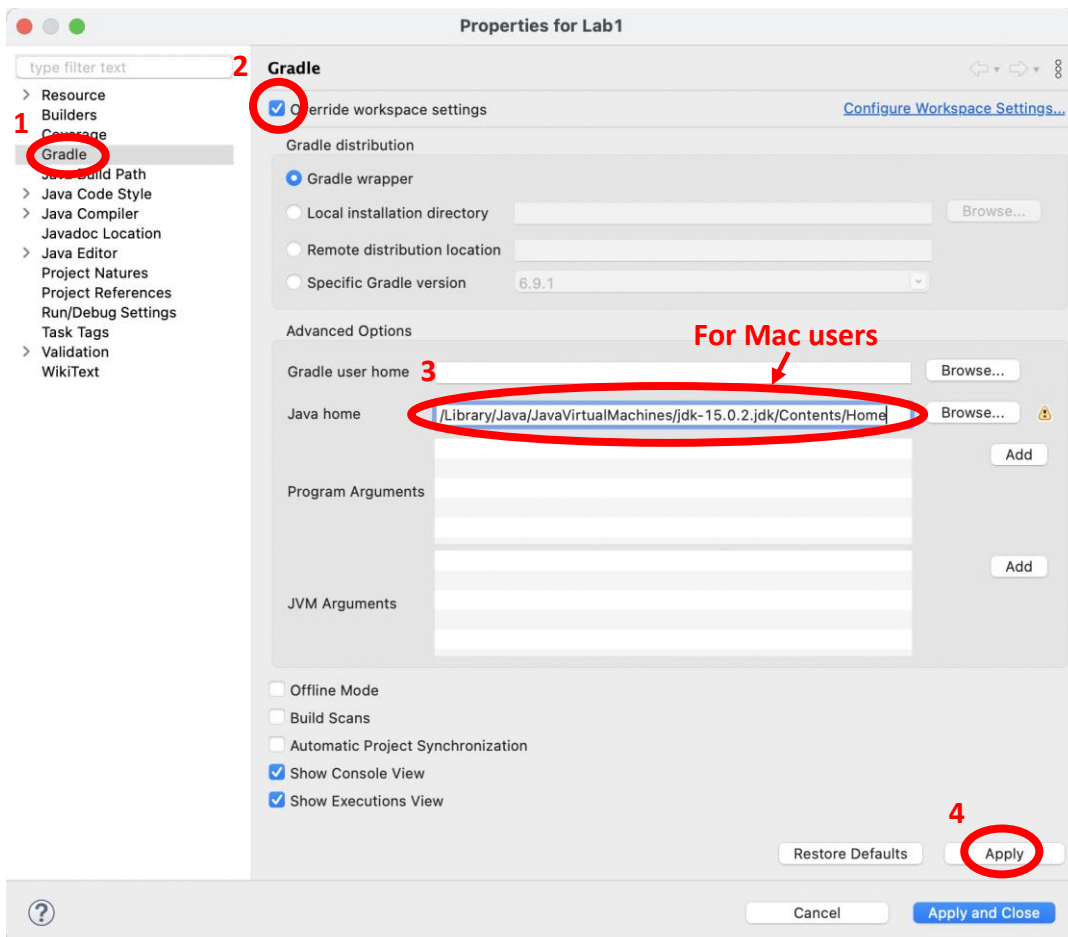
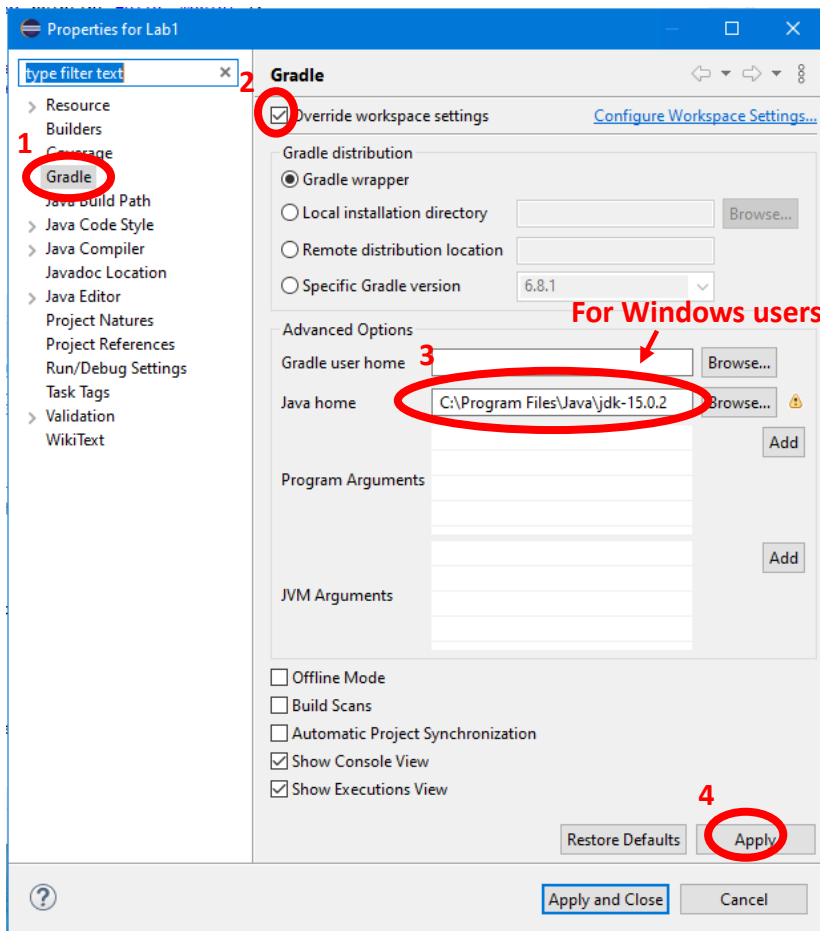
Step 1.5: Open Library.java and add a function



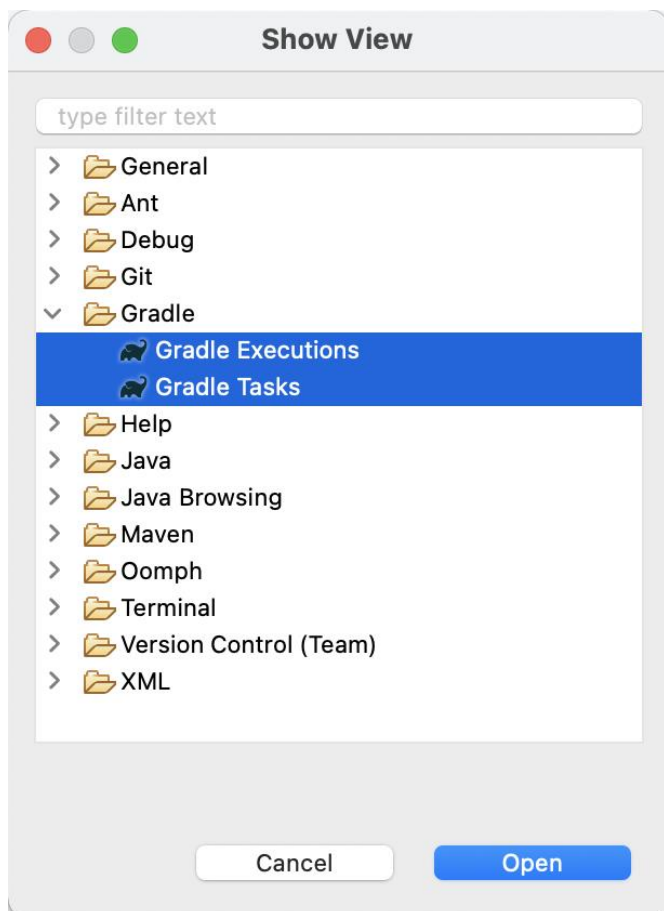
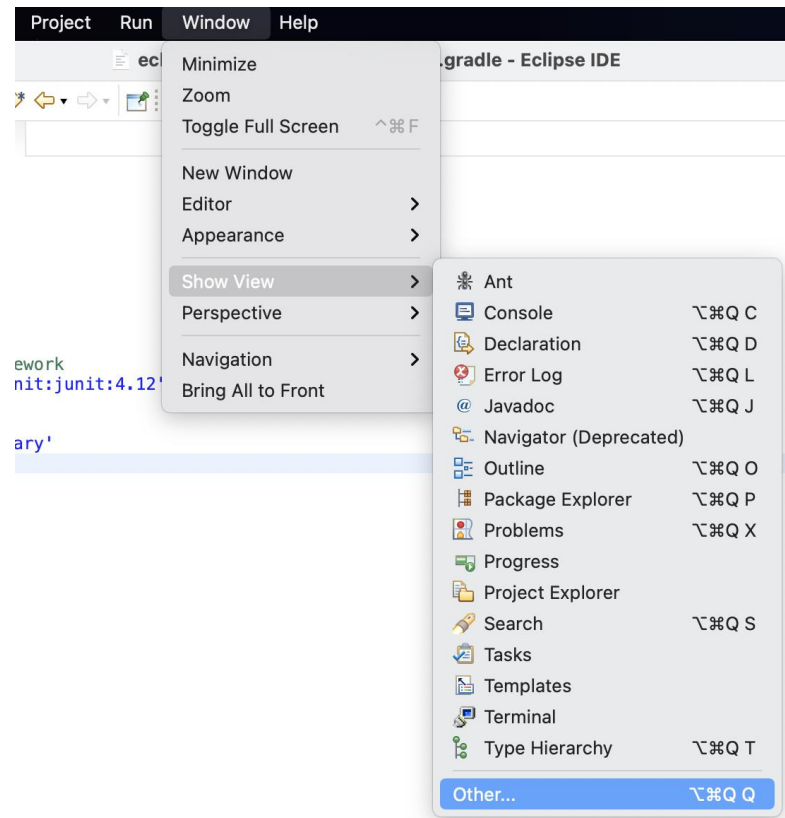
Step 1.6: Edit the file build.gradle. Make sure you have save both Library.java and build.gradle.



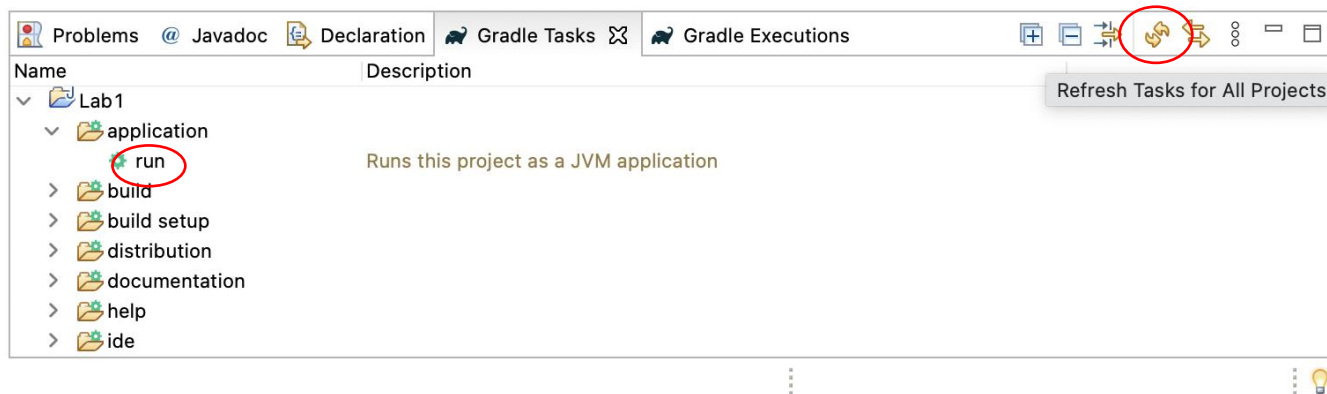
Step 1.7: Click “Project > Properties” from the menu bar, and select “Gradle”. Check the box next to “Override workspace settings” and set the “Java home” to the JDK installed folder as shown.



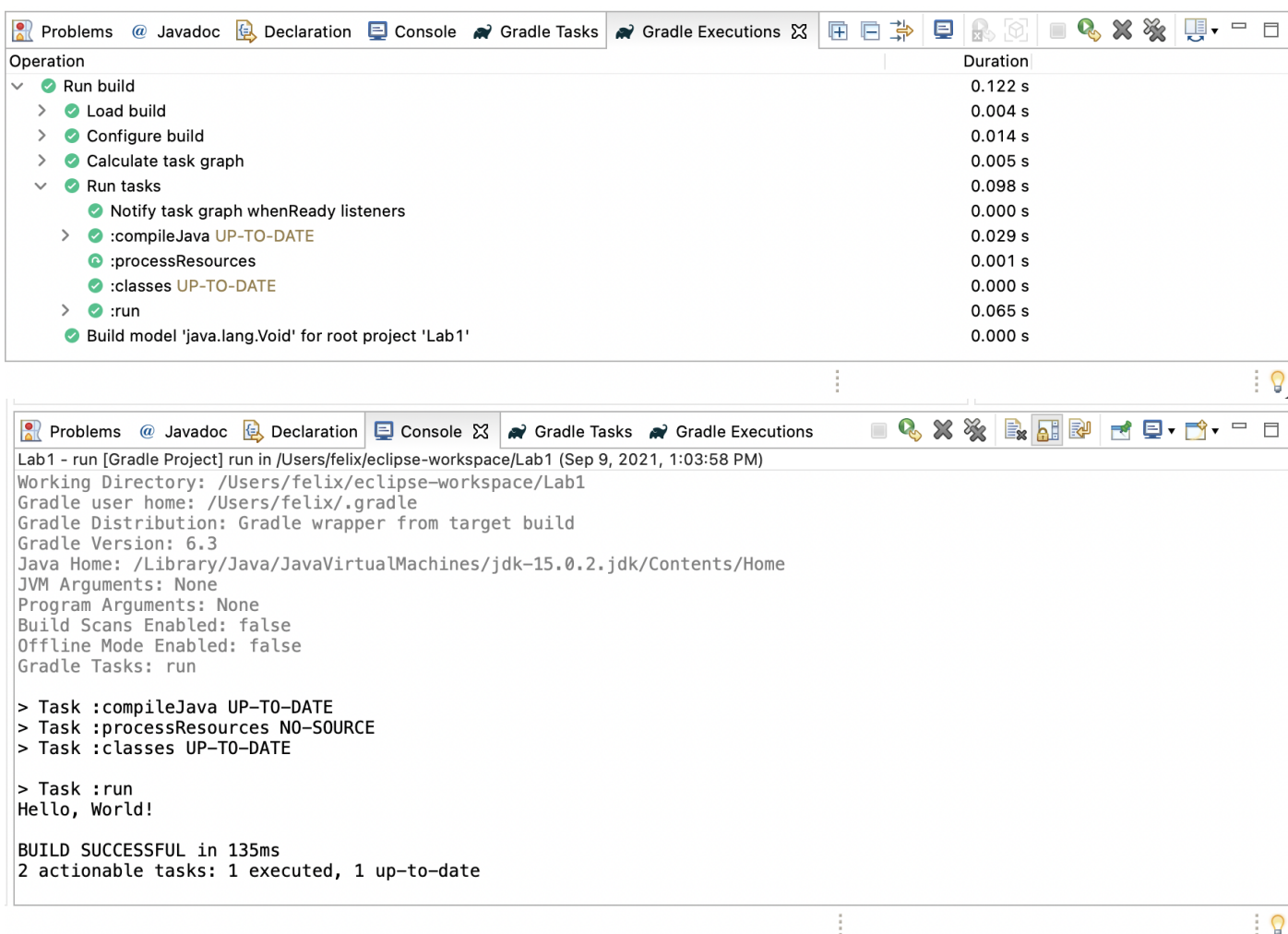
Step 1.8: Click “Windows > Show View > Other” from the menu bar. Open both Gradle Executions and Gradle Tasks. (Note: you can use the shift key on your keyboard to select multiple items at a time)



Step 1.9: On the tab “Gradle Tasks”, (double) click “Application > Run”. If you cannot find Application, click the refresh button and do it again.

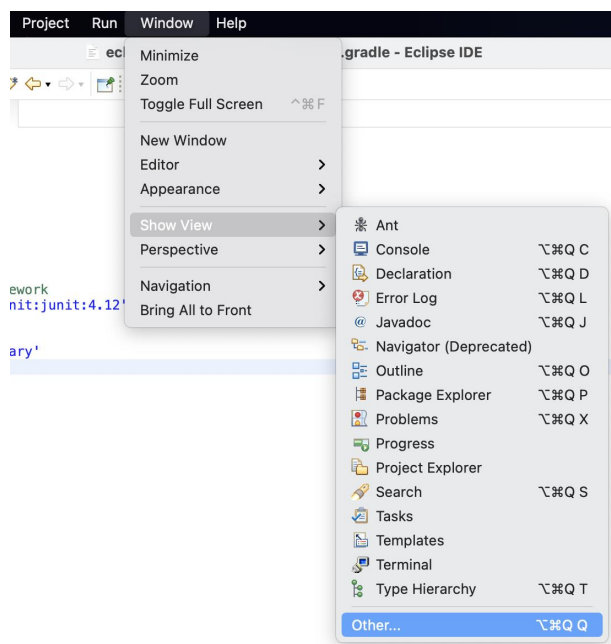


Step 1.10: Check your Gradle Executions tab and your Console tab. They should look like:



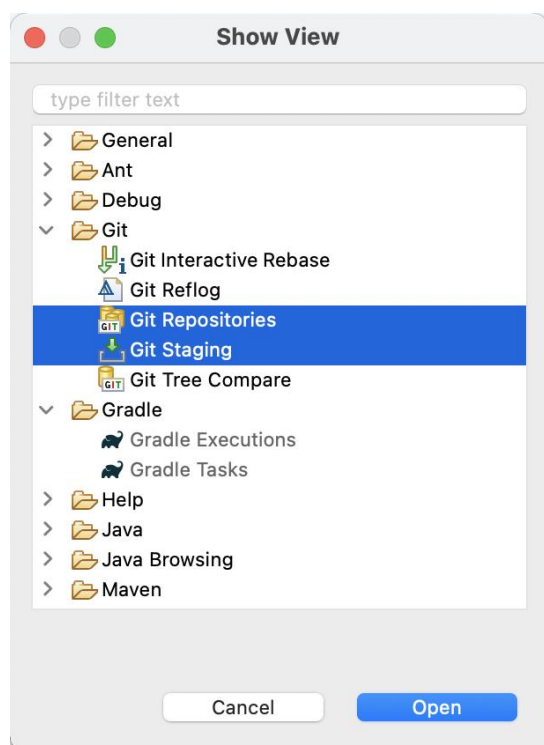
Exercise 2: Setup a local Git repository

Step 2.1: From the menu bar, select “Window > Show View > Other...”



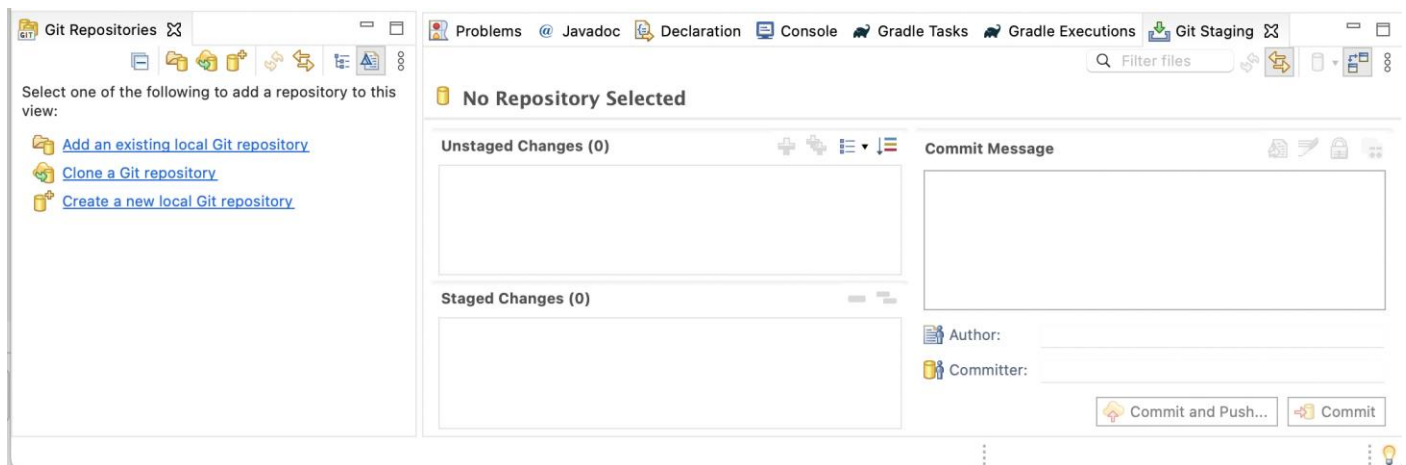
Step 2.2: Select “Git Repositories” and “Git Staging”. After that, click “Open”

Hint: You can click the “SHIFT” key to select multiple items

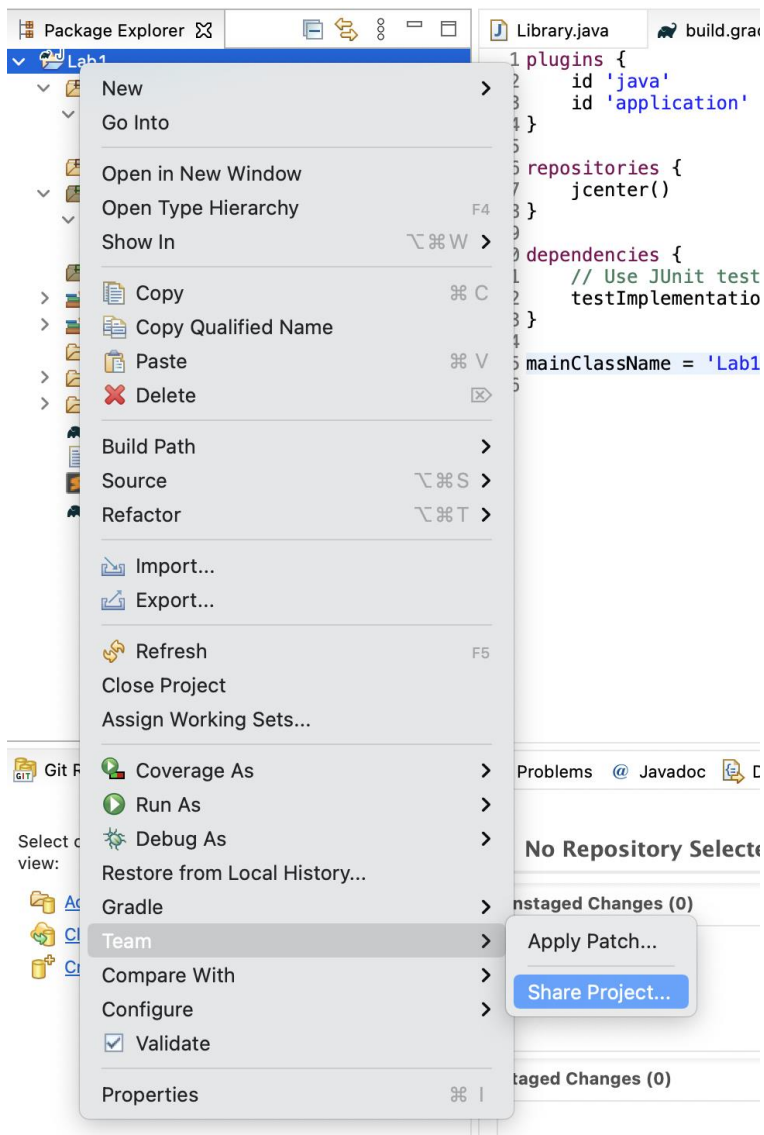


Verification: You are expected to see the following screenshot:

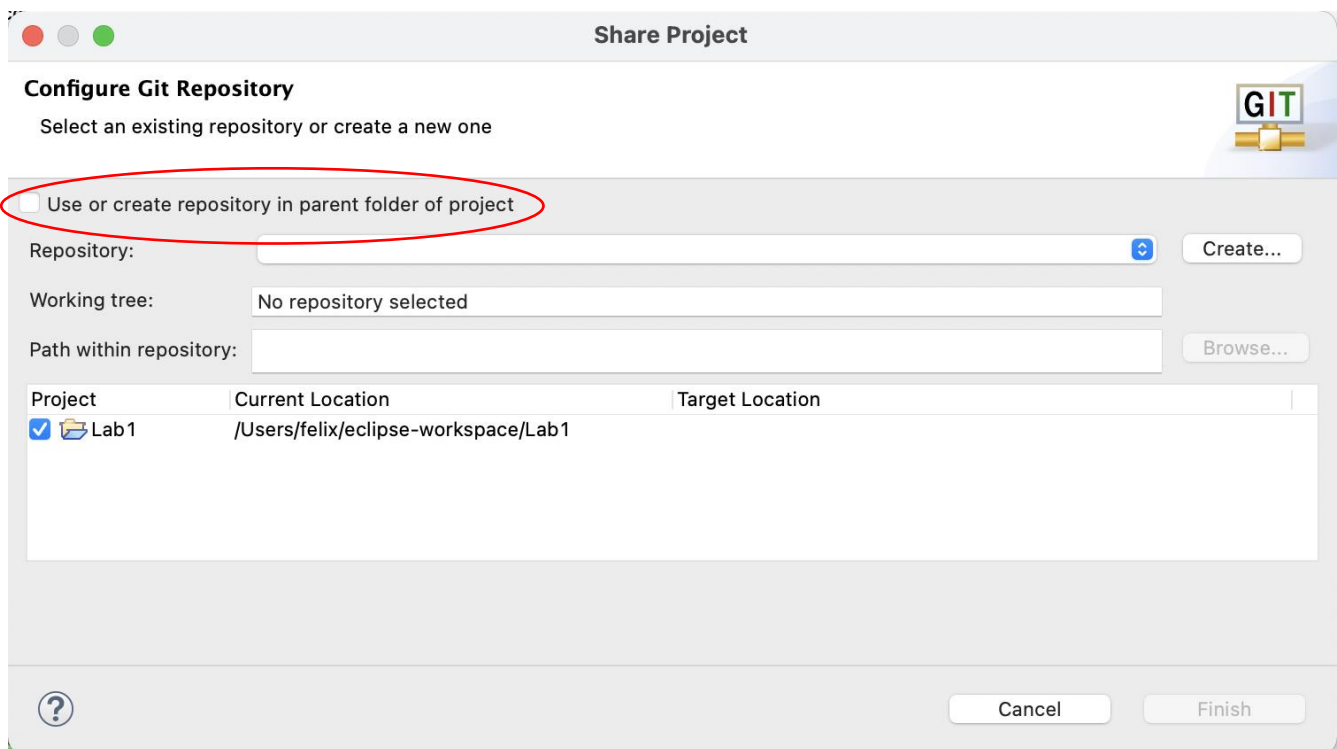
Note: If you accidentally close “Git Repositories” and “Git Staging”, follow the above steps to restore the windows



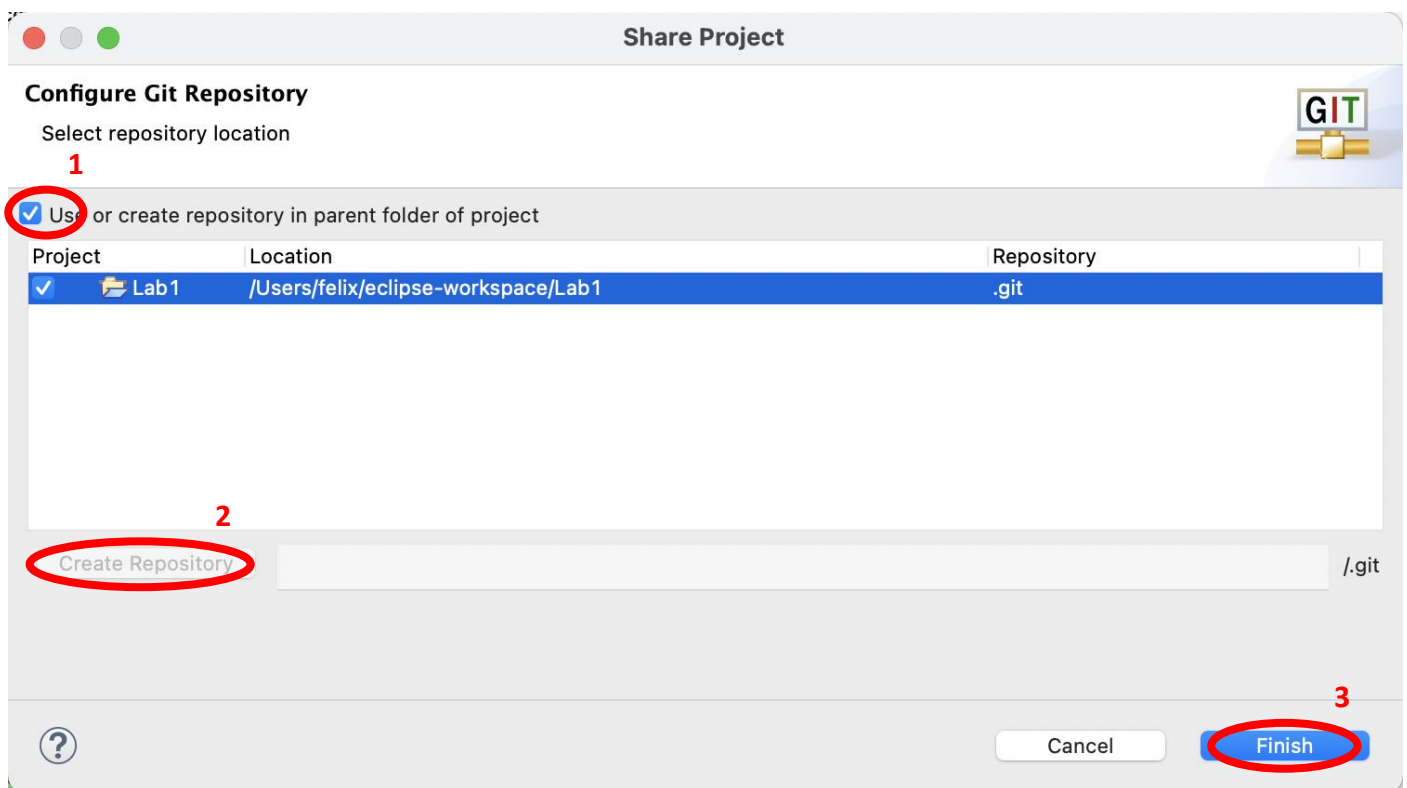
Step 2.3 Right-click the project folder, select Team > Share Project...



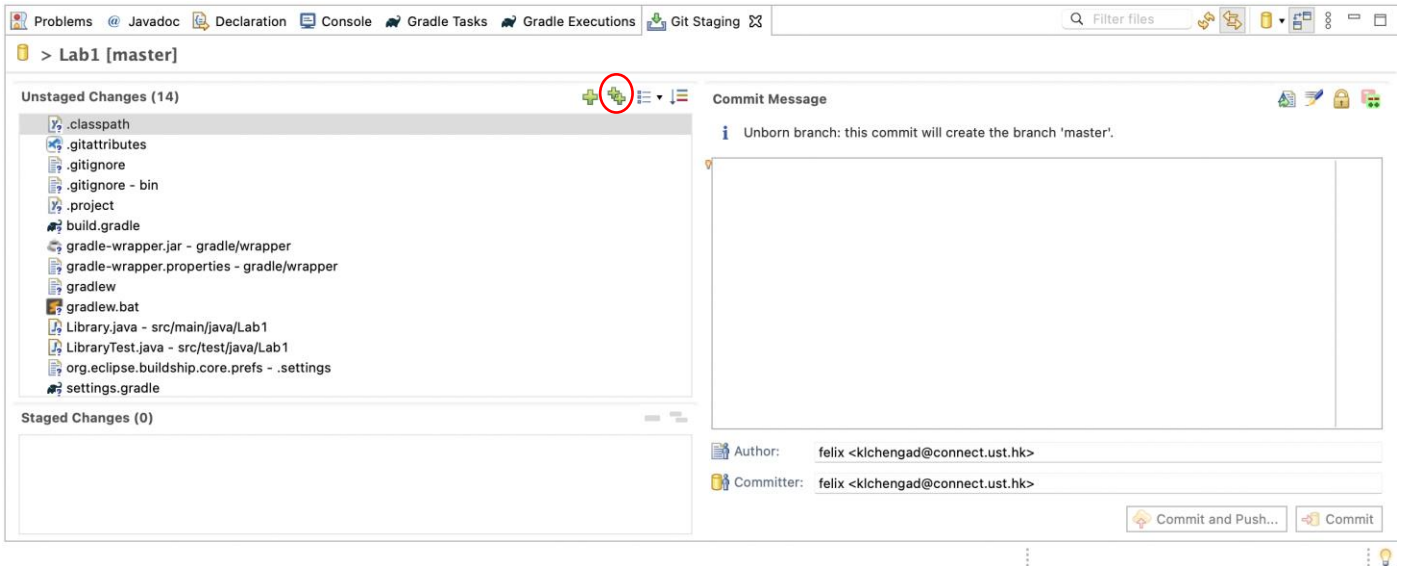
Step 2.4: First, check the “Use or create repository in parent folder of project”. Second, click “Create Repository”. A hidden “.git” folder will be created in the project directory with some configuration files. After that, you can click “Finish” button



Note: You must click “Create Repository” button first. Otherwise, the “Finish” button will be disabled.



Step 2.5: Select Library.java from the Package Explorer. Compare your “Git Staging” windows with the screenshot below:

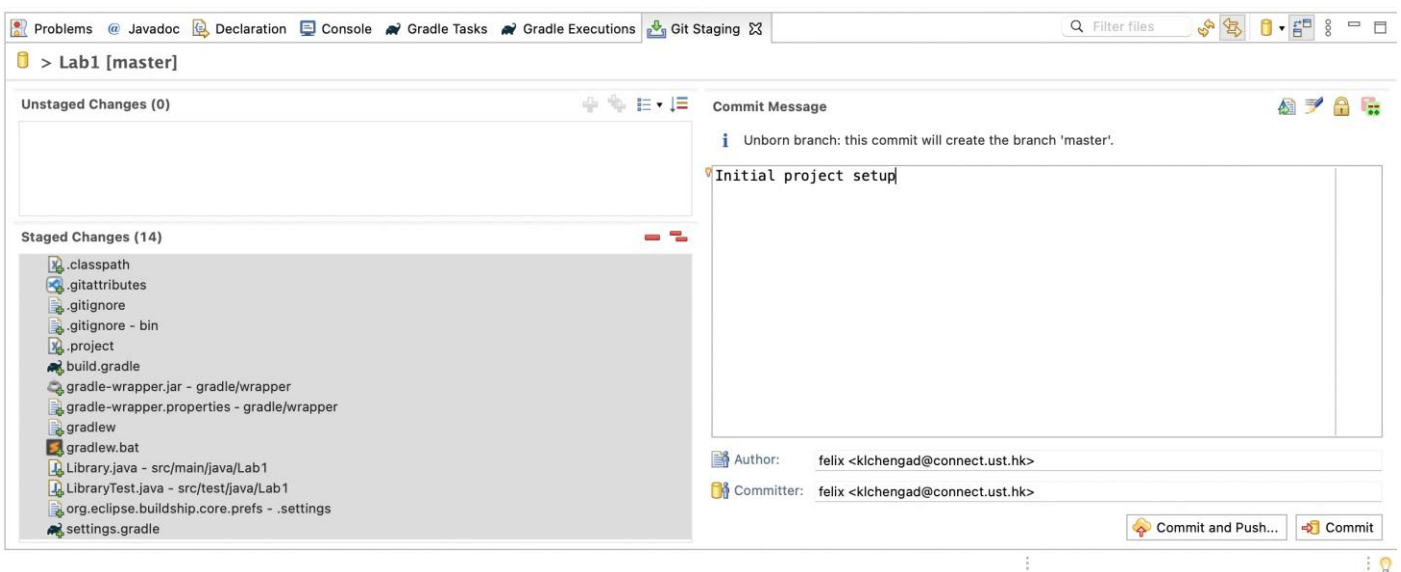


Further explanation:

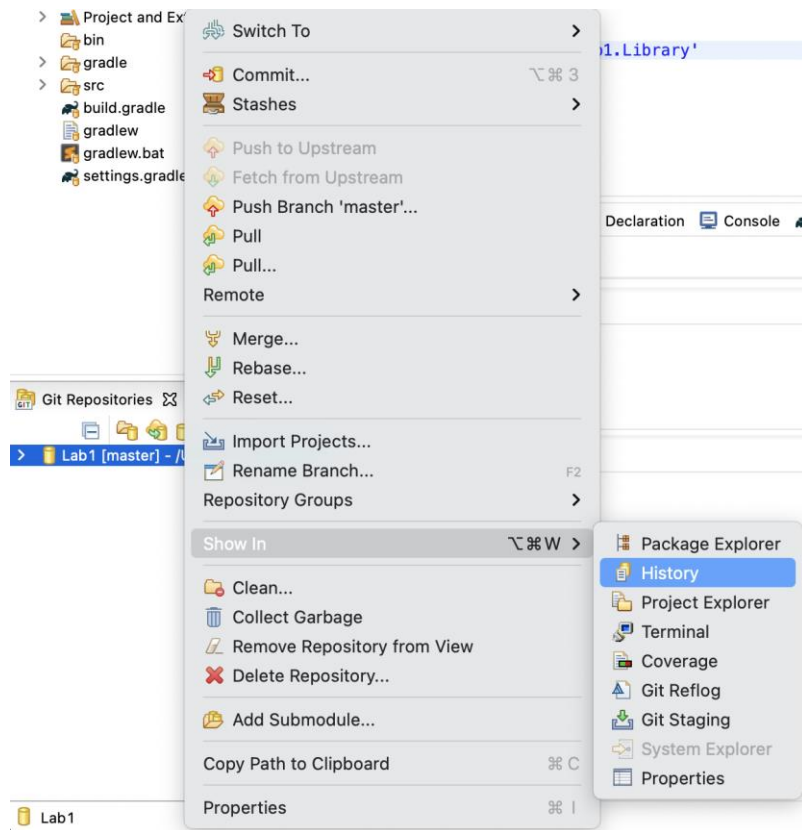
- *Un-staged changes:* Files that are already changed, but not yet ready to be tracked by the repository
- *Staged Changes:* Files that are changed and be ready to be pushed to and tracked by the repository
- *Sample usage:* For example, you modified 10 files. You are 100% confident on the changes made in the first 9 files, but not very confident about the 10th file. In the next commit, you can stage the first 9 files, and not to stage the 10th file.

Step 2.6: Click “Add all files” button (as circled in the above screenshot) on the “Unstaged Changes” tab of the “Git Staging” window. All files should now be staged. Type in a meaningful commit message (e.g. Initial project setup). After that, click the “Commit” button.

Further explanation: Developers should always write a short, precise, and meaningful commit message. Otherwise, it will be hard for other developers to understand the commit log.



Step 2.7: From “Git Repositories”, right click the repository and select “Show In > History”



Verification: The following commit log message should be displayed in the History window

Problems @ Javadoc Declaration Console Gradle Tasks Gradle Executions Git Staging History

Repository: Lab1

Id	Message	Author	Authored Date	Committer	Committed Date
27404f5	master (HEAD) Initial project setup	felix	2 minutes ago	felix	2 minutes ago

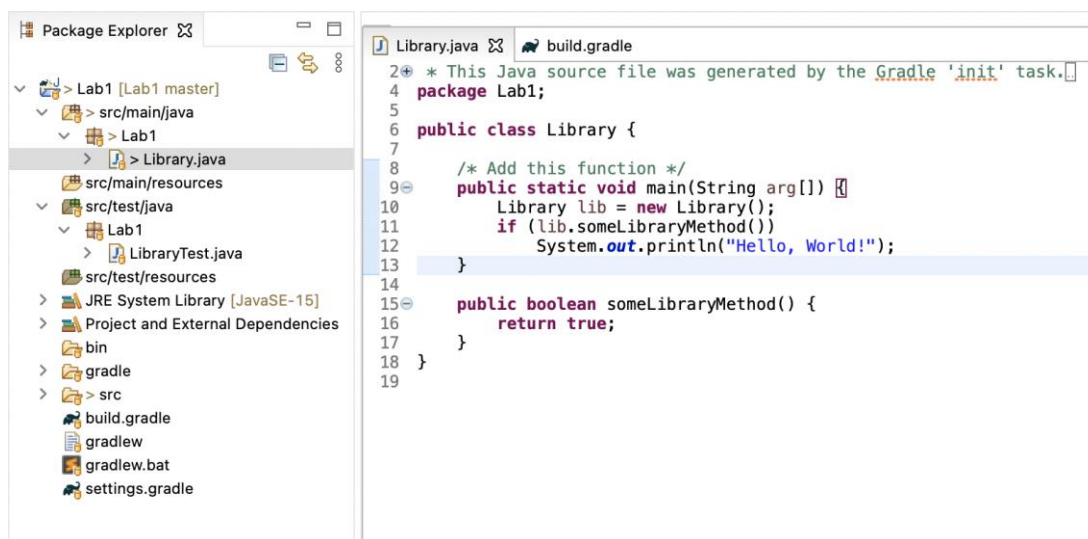
commit 27404f5353c65afd8d5c1de0a9d48e73f09d6fa3
Author: felix <klchengad@connect.ust.hk> 2021-09-09 13:31:35
Committer: felix <klchengad@connect.ust.hk> 2021-09-09 13:31:35
Branches: **master**

Initial project setup

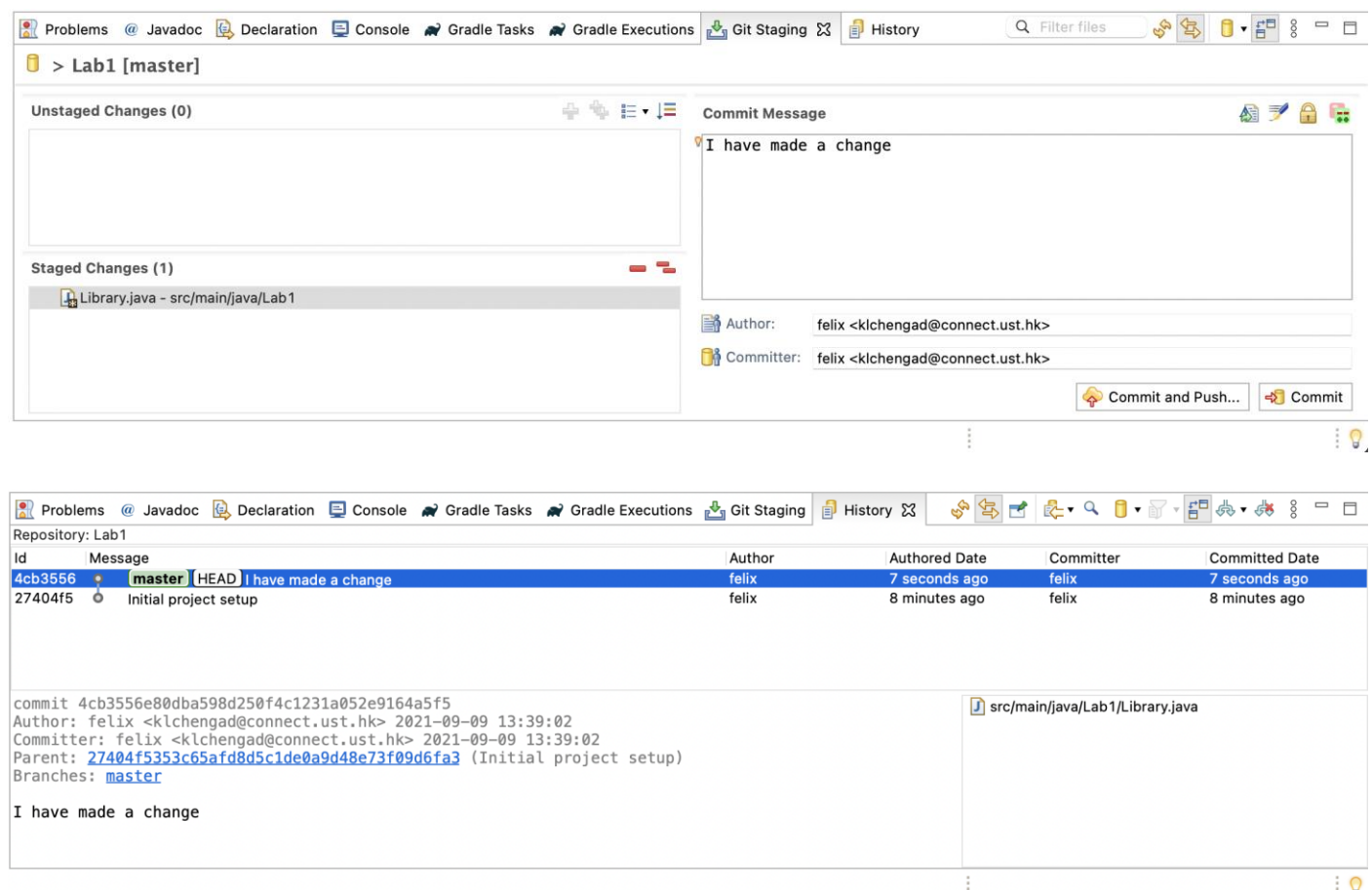
- .classpath
- .gitattributes
- .gitignore
- .project
- build.gradle
- gradlew
- gradlew.bat

Exercise 3: Make and commit changes to the repository

Step 3.1: Modify Library.java as follows.



Step 3.2: At your Git Staging Tab, stage Library.java (by clicking the add button). Type in a meaningful commit log message and then click “Commit”.



Exercise 4: Create a GitHub account and track changes

Step 4.1: Register/Sign in your GitHub account

GitHub (<https://www.github.com>) is a web-based hosting service for version control using git. You should register for a GitHub Free account to complete this lab assignment (as well as subsequent lab assignments and the team project). GitHub Free enables developers to host public/private repositories with unlimited collaborators at no cost.

After you have created your GitHub Free account. It is good to know that GitHub Education account is also available for HKUST students. You could apply for it with more advanced features for better experience. Check out the details here: <https://education.github.com/students>. You should use your <itsc@ust.hk> email (without connect) for application instead of your other email accounts.

What e-mail address do you use for school? *

Note: Selecting a school-issued email address gives you the best chance of a speedy review.

☒ klchengad@ust.hk

✓ The Hong Kong University of Science and Technology (HKUST)

+

 Add an email address

What is the name of your school? *

Note: If your school is not listed, then enter the full school name and continue. You will be asked to provide further information about your school on the next page.

The Hong Kong University of Science and

We chose this school based on your email. If this isn't your school, please use a different email.

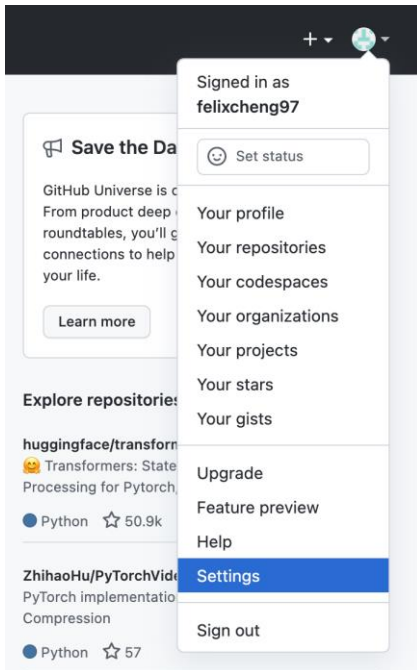
How do you plan to use GitHub? *

Please note, your request cannot be edited once it has been submitted, so please verify your details for accuracy before sending them to us.

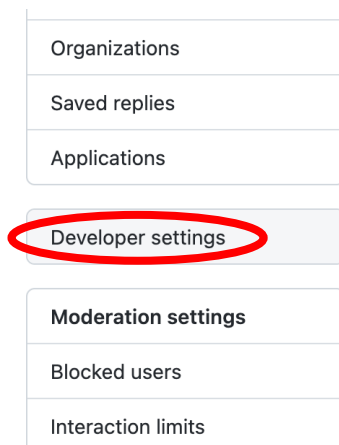
Continue

We recommend students to use GitHub. There are other web-based Git services (e.g. GitLab, BitBucket). You may need to self-study the setup instructions for these alternative services.

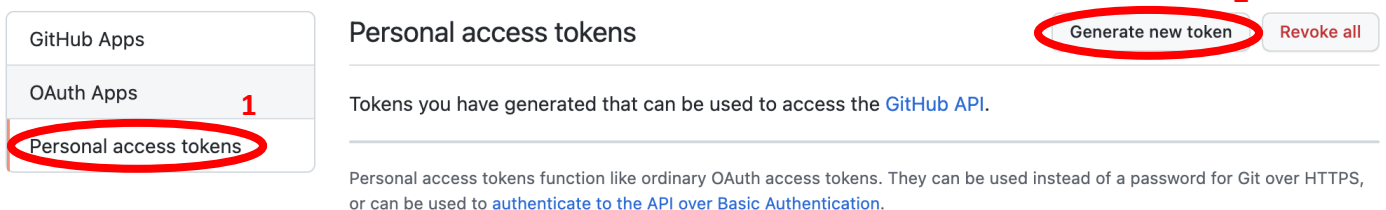
Step 4.2: Sign in with your GitHub account (in this example, the user ID is “felixcheng97”). In the upper-right corner, click your profile photo, then click “Settings”.



Step 4.3: In the left sidebar of the prompt page, click “Developer settings”.



Step 4.4: In the left sidebar of the prompt page, click “Personal access tokens”. Then click “Generate new token”.



Step 4.5: Give your token a descriptive name, which can be anything you like (in this example, we use “comp3111”). Then set expiration date using “Custom...” mode as “12/31/2021”. Then tick the “repo” scope for this token.

GitHub Apps

OAuth Apps

Personal access tokens

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note 1

comp3111

What's this token for?

Expiration * 2

Custom... 12/31/2021

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input type="checkbox"/> workflow	Update GitHub Action workflows

☐ admin:gpg_key Full control of public user GPG keys ([Developer Preview](#))

☐ write:gpg_key Write public user GPG keys

☐ read:gpg_key Read public user GPG keys

Generate token [Cancel](#) 3

After you have done all the things above, scroll to the very bottom and click “Generate token”.

Step 4.6: Copy the token string to a safe place for later use. Please treat your tokens like passwords and keep them secret. You can treat the token as a more secured password with expiration date for your GitHub account with specific granted permissions.

Personal access tokens

[Generate new token](#) [Revoke all](#)

Tokens you have generated that can be used to access the [GitHub API](#).

Make sure to copy your personal access token now. You won't be able to see it again!

✓ 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 [Delete](#) Copy this and save it somewhere safe.

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Step 4.7: On your GitHub, click “+” button at the top-right corner to create a “New Repository”. Name the repository and click “Create repository”

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Owner *

Repository name *



felixcheng97



comp3111-lab1-2021f



Great repository names are short and memorable. Need inspiration? How about [legendary-spoon](#)?

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ **Add a README file**

This is where you can write a long description for your project. [Learn more](#).

☐ **Add .gitignore**

Choose which files not to track from a list of templates. [Learn more](#).

☐ **Choose a license**

A license tells others what they can and can't do with your code. [Learn more](#).

Create repository

Step 4.8: In the next page, select “HTTPS” and copy the link circled.

Quick setup — if you've done this kind of thing before



Set up in Desktop

or

HTTPS

SSH

<https://github.com/felixcheng97/comp3111-lab1-2021f.git>



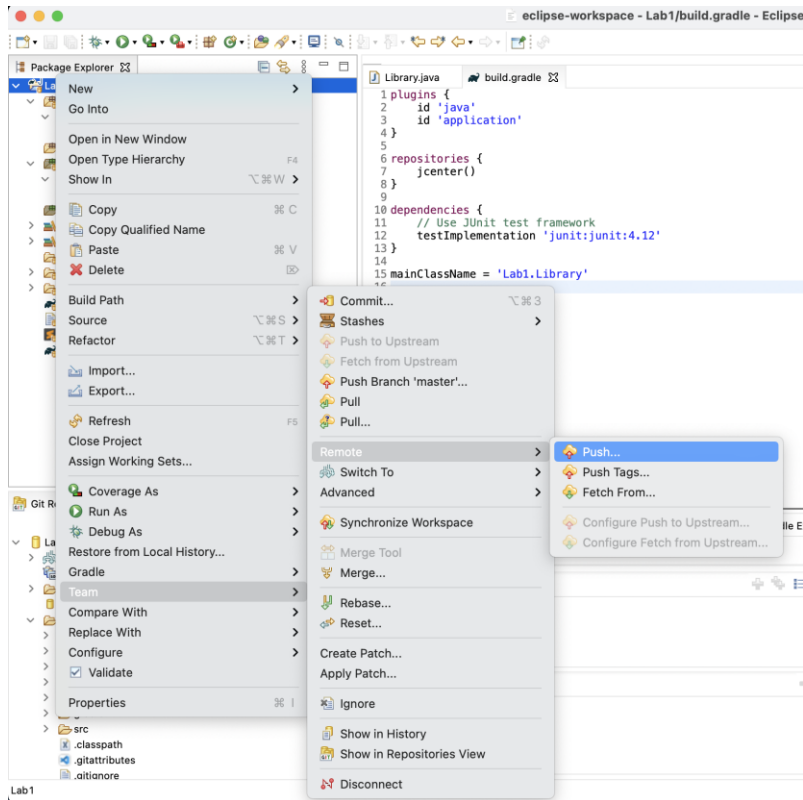
Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

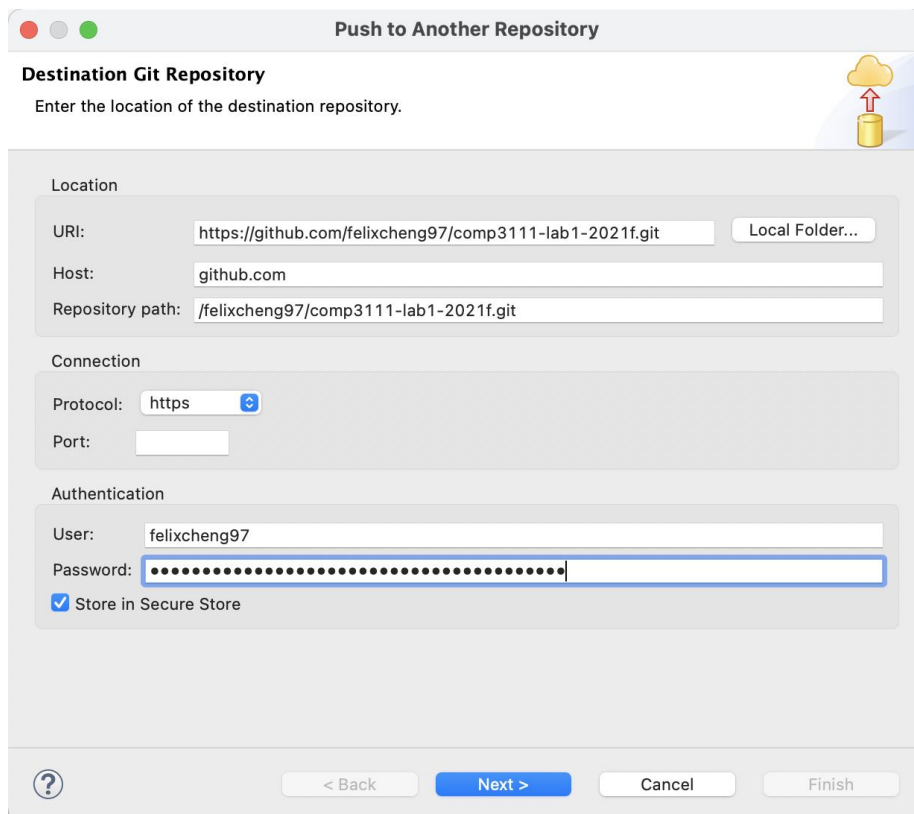
```
echo "# comp3111-lab1-2021f" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/felixcheng97/comp3111-lab1-2021f.git
git push -u origin main
```



Step 4.9: Back to Eclipse and right-click the project folder. Select “Team > Remote > Push...”.



Step 4.10: Paste the https link for the GitHub repository in the box “URI”. Select “https” protocol. Enter your GitHub account in “User”; copy and paste your generated token in “Password”. Check the “Store in Secure Store” box and click “Next”.



Step 4.11: Select the master branches from source and destination ref. Click “Add All Branches Spec” (to enable the “Finish” button). After that, click the “Finish” button.

Push to: <https://github.com/felixcheng97/comp3111-lab1-2021f.git>

Push Ref Specifications

Select refs to push.

Add create/update specification

Source ref:

refs/heads/master

Destination ref:

refs/heads/master

+ Add Spec

Add delete ref specification

Remote ref to delete:

X Add Spec

Add predefined specification

Add Configured Push Specs

Add All Branches Spec

Add All Tags Spec

Specifications for push


Mode	Source Ref	Destination Ref	Force Update	Remove

Force Update All Specs


Remove All Specs


? < Back Next > Cancel Finish


Step 4.12: Reload your GitHub webpage. Verify that your project has successfully been linked to a remote repository.

 **felixcheng97 / comp3111-lab1-2021f** Public

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

 master ▾


 1 branch













 0 tags

[Go to file](#)

[Add file ▾](#)

[Code ▾](#)

felix I have made a change 4cb3556 5 hours ago  2 commits

 .settings	Initial project setup	5 hours ago
 bin	Initial project setup	5 hours ago
 gradle/wrapper	Initial project setup	5 hours ago
 src	I have made a change	5 hours ago
 .classpath	Initial project setup	5 hours ago
 .gitattributes	Initial project setup	5 hours ago
 .gitignore	Initial project setup	5 hours ago
 .project	Initial project setup	5 hours ago
 build.gradle	Initial project setup	5 hours ago
 gradlew	Initial project setup	5 hours ago
 gradlew.bat	Initial project setup	5 hours ago
 settings.gradle	Initial project setup	5 hours ago

Help people interested in this repository understand your project by adding a README.

[Add a README](#)

Lab Assignment and Assessment

Lab Assignment:

1. Add a textfile "readme.md" to your project folder in Eclipse
2. Put a pseudo name and any 8-digit number in the file
3. Commit with the message: "Added a readme.md file"
4. Push it to GitHub
5. Delete the file src/test/java/LibraryTest.java in Eclipse
6. Commit with the message: "Deleted LibraryTest.java"
7. Take a screenshot of the "Git History" panel in your Eclipse to reflect the changes made
8. Add the screenshot file to your project folder in Eclipse
9. Find a way to link the screenshot image in readme.md (to be shown on GitHub)
10. Commit your screenshot file and readme.md, and push them to GitHub
11. Find a way to locate the deleted LibraryTest.java on GitHub

Note:

- *To take a screenshot on Windows, press PrtSc button on your keyboard. To do it on Mac, press cmd-shift-3 and the screenshot file will be saved on your desktop.*
- *The screenshot file should show four commits. But your GitHub may contain more than that.*
- *Do some research to find out how to insert an image in readme.md (Keyword: "markdown image")*

Assessment

1. The front page of your GitHub should look like this:

The screenshot shows the GitHub interface for a repository named 'felix A final push to completion!'. The top bar indicates 'master' branch, '1 branch', and '0 tags'. Buttons for 'Go to file', 'Add file', and 'Code' are visible. The file list shows various project files and folders, including '.settings', 'bin', 'gradle/wrapper', 'src/main/java/Lab1', '.classpath', '.gitattributes', '.gitignore', '.project', 'build.gradle', 'gradlew', 'gradlew.bat', 'image.png', 'readme.md', and 'settings.gradle'. The commit history for 'readme.md' is shown below, listing three commits by 'felix'.

Id	Message	Author	Authored Date	Committer	Committed Date
5f622a2	Deleted LibraryTest.java	felix	1 seconds ago	felix	1 seconds ago
1293a2c	Added a readme.md file	felix	49 seconds ago	felix	49 seconds ago
4cb3556	I have made a change	felix	6 hours ago	felix	6 hours ago
27404f5	Initial project setup	felix	6 hours ago	felix	6 hours ago

2. There should be no LibraryTest.java on your GitHub folder src/test/java (deleted already). However, you should be able to locate the deleted LibraryTest.java in some earlier versions of the repository being maintained at GitHub.

Submission

Make an online submission of your lab work (two personalized URLs) through Canvas by the due date:

- URL link to your repository on GitHub
- URL link to the deleted LibraryTest.java on GitHub (in earlier versions of your repository)