CS5004, Spring 2024

Lab1: Setting Up and Getting Started

Therapon Skoteiniotis[1], Tamara Bonaci and Abigail Evans
skotthe@ccs.neu.edu, t.bonaci@northeastern.edu, a.evans@northeastern.edu

1. Summary

In today's lab, we will:

- Configure our machines to use Java and IntelliJ
- Show how to create a Gradle project, and update build.gradle file
- Walk through a simple example to create a class in Java
- Practice designing simple classes
- Practice designing classes that contain other classes

Note 1: Labs are intended to help you get started, and to give you some practice while the course staff is present, and able to provide assistance. You are not required to finish all the problems in this lab assignment, but you are expected to push your lab work to your individual repo on the GitHub course organization.

2. Setup

We will be using **Java 21, IntelliJ and Gradle.**

You can download and install Java 21 from here:
https://www.oracle.com/java/technologies/javase/jdk21-archive-downloads.html

The instructors will be using the latest version of IntelliJ in class and labs.

- IntelliJ (https://www.jetbrains.com/idea).
    1. As a student you, get the full version of the IDE for free (https://www.jetbrains.com/student/). Apply for the student discount first.
    2. Download IntelliJ Ultimate Edition (https://www.jetbrains.com/idea/download)

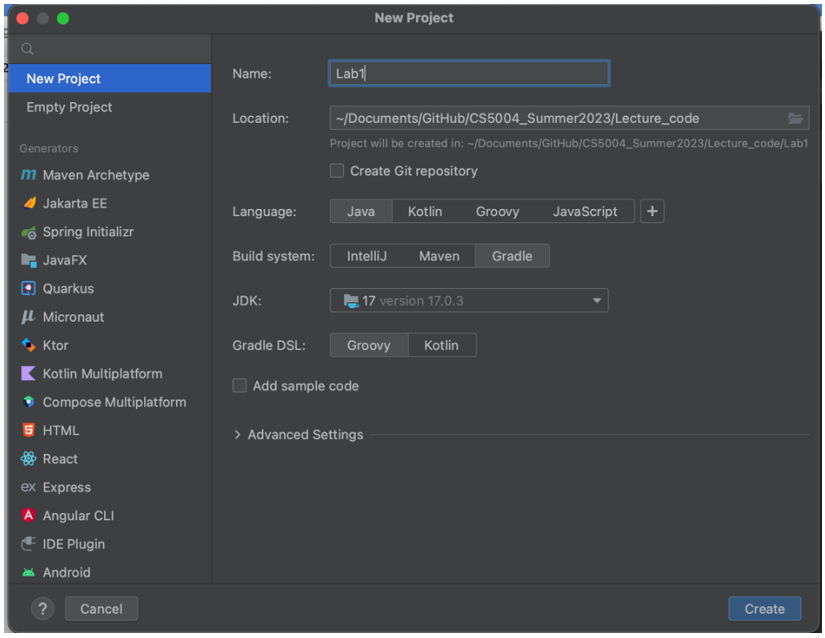
[1] Assignment modified from the original version prepared by Dr. Therapon Skoteiniotis.

3. Install IntelliJ on your machine using the process that you typically use for your operating system to install new software

Note 2: If you select an IDE other than the one used in class, we will do our best to help you, but please be prepared to do extra work on your own. We will not give you extra time or amend your grade due to any possible issue that you might face with your IDE's configuration.

3. Creating Your First Gradle Project

- Open IntelliJ
- Either from IntelliJ's main menu select `New → Project`, or from the "Welcome" dialog, you can choose `New Project`.
- Select **Gradle** for the Build System and keep the default **Groovy** for the Gradle DSL
- The JDK will likely auto select an installed JDK. If you've only installed 21, it should find it. If not, you can go through the "Add JDK" to add it. If something other than 21 was selected, please make sure you select 21.



- You should see two fields:
  a. **"Name"**, provide a name for your project. This name can be anything you like, but today, you probably want it to be Lab1.
  b. **"Location"**, this is the location on your computer that IntelliJ is going to use to create and store all your code. There should be a button to the far right of this field whose title is three dots, i.e., `...`.
- Click `Create`
- IntelliJ will now open in a new window that will contain two tabs:

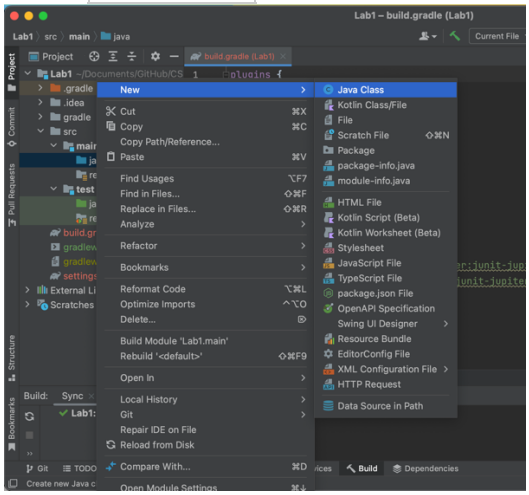
a. A left tab that is your **Project Explorer**. This is similar to your file explorer.

b. A right tab, the **Editor**, that will be used to show the contents of files that you select in the Project Explorer.

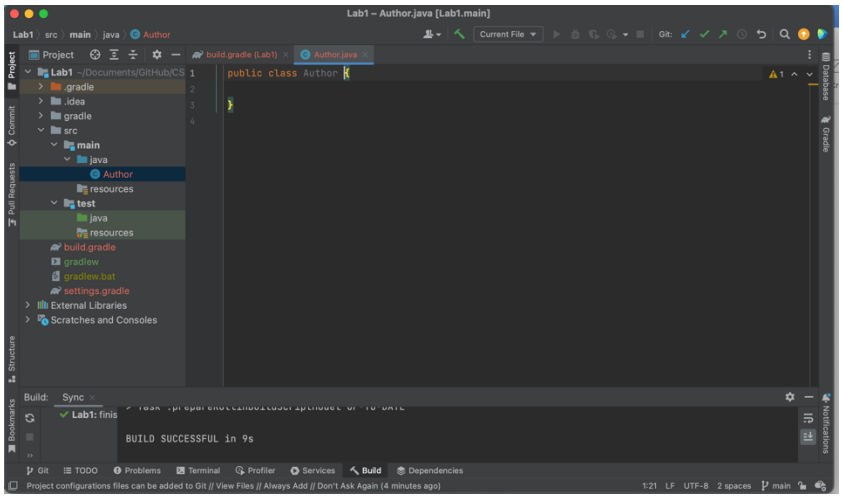
- In the Project Explorer you should see a folder with the name you gave to your project. Double click on the folder's name to expand it if it isn't already expanded.

4. Creating Your First Java Class

- You should see a sub-folder named **src\main\java** (you will notice IntelliJ may take a few moments to generate all the folders).
- Right click on the folder named java to open the context menu. From the context menu select `New → Java Class` to create a new Java Class.



- A small window will pop-up asking you to provide a name for your class. Input the name **Author** and click `OK` or press enter.
- The right tab of your IntelliJ window should now contain a minimal Java class with an empty Java class definition:

- 

- Notice that in the Project Explorer tab there is now a new file named Author under the folder named src.
- To test that your setup is working as expected, replace all the contents of the file Author.java with the following code:

Author.java

```
/**
 * Represents an Author with their details--name, email and physical address
 *
 * @author therapon
 *
 */

public class Author {

 private String name;
 private String email;
 private String address;

 /**
  * Creates a new author given the author's name, email and address as strings.
  *
  * @param name the author's name
  * @param email the author's email address
  * @param address the authors physical address
  */
 public Author(String name, String email, String address) {
  this.name = name;
  this.email = email;
  this.address = address;
 }

 /**
  * @return the name
```

```
 */
 public String getName() {
  return this.name;
 }

 /**
  * @return the email
  */
 public String getEmail() {
  return this.email;
 }

 /**
  * @return the address
  */
 public String getAddress() {
  return this.address;
 }
}
```

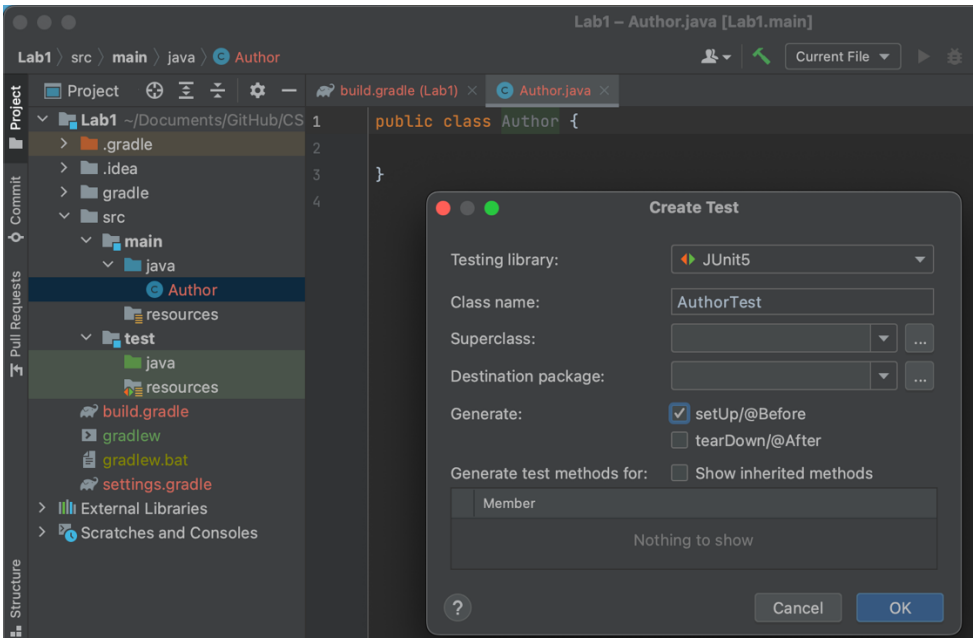
5. Creating Your First Java Test Class

IntelliJ tries to assist you while you code by popping up an image of a small light bulb inside your editor (the right tab). Move your cursor to be inside the name of class, i.e., the word Author in the class header public class Author. Wait for a couple of seconds and the yellow light bulb image should appear at the start of that line. You can force the IntelliJ assistant to open using the following keystrokes

- Windows/Linux : Alt + Enter
- Mac : Option + Enter

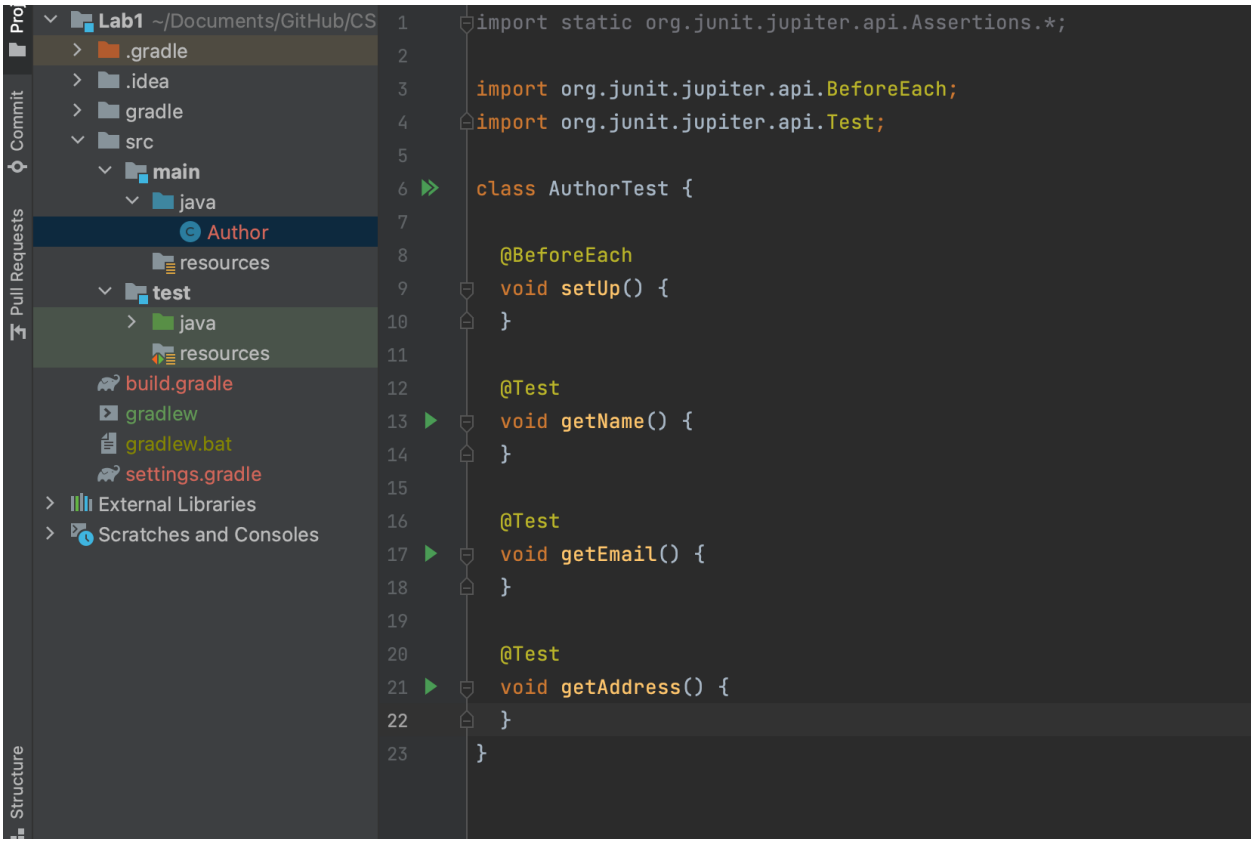
Click on the yellow light bulb icon and a menu should appear.

From the menu select Create Test. This action will cause a new pop-up window to appear.



In the new pop-up window, we will configure and create a test for our `Author` class. Starting from the top of the window going down:

- For **"Testing library"** select `JUnit 5`
- Leave **"Superclass"** empty.
- Leave **"Destination Package"** empty
- Select the check mark with the title **"setUp/@Before"** only.
- At the bottom of this pop-up window you should see the list of methods that are available in class `Author` for testing. Select **all** of them.
- Click `OK`
- Select `JUnit Test Case` from the pop-up menu. This will create a new Java class (and a new file) called AuthorTest.
- The Package Explorer should now have a new file with the name `AuthorTest` under the folder `src`
- The editor should now display something like the class presented below:



To run our tests:

- press the green "play" button at the top of the IntelliJ window
- or press the green "play" button on the **left** margin on the line that contains the Java class definition header, e.g., `public class AuthorTest {`

IntelliJ will run your tests and the results of the test run are displayed in a new tab that opens at the bottom of the IntelliJ window.

(Please notice that the tests are not yet implemented).

6. Introduction to Git and GitHub

Prerequisite: for this part of the lab, you will need your **Khoury account**. If you don't already have a Khoury account, please create one using the following link: https://my.ccs.neu.edu/account/apply

6.1. Introduction

Git is a widely used, open source software for file management and version control. GitHub is an online code hosting platform that includes Git version control. For our course, we will be using the Khoury Enterprise GitHub (a private GitHub instance) to submit homework and lab assignments.

The URL to the course organization on Khoury GitHub is:

https://github.ccs.neu.edu/cs5004-sea-summer2021/lecture_material

On the course organization, you should have access to two repositories (repos):

- Repo accessible to all students, named **`lecture-code`**
- Repo accessible to you only

You will need to access your individual repo, and clone it, in order to connect a working directory on your own machine (local copy) with your repo on remote server.

6.2. Using Khoury GitHub

When it comes to using git on your machines, you have many different options, such as:

- the Git command line tools,
- the GitHub GUI for Mac, or
- the GitHub GUI for Windows.

We will show how to use git on command line, and how to use GitHub GUI, but this semester, **we recommend using the command line**. Here are the instructions on how to get started with command line:

Connecting to the Khoury GitHub server Using Command Line

- **Step 1: Clone your personal repository**
  - You only need to complete this step once. If you've already cloned your repository, skip to step 3.
  - First, find the URL for your personal repository. In your browser, login to Khoury GitHub and open your personal repository. Click the *Clone or download* button and copy the URL.
  - Open the **terminal** or equivalent. Navigate to the local folder where you want to store your files by typing:

```
cd <path to folder>
```

    - <path to folder> is the full path of the folder on your machine.

        - cd stands for “change directory”.

    - Then, type the following:

```
git clone <remote repo URL>
```

        - <remote repo URL> is the URL your copied earlier.
        - Hit Return/Enter. You may be prompted to login to Khoury GitHub. Once the repo has been cloned, cd into the newly created repo folder:

```
cd <folder name>
```

- **Step 2: Fetch changes from the remote repository**
    - This step only applies once your repository is active. Open Terminal or equivalent, and navigate to your local repository. Type:

```
git pull
```

- **Step 3: Create a folder for the assignment**
    - If you are just getting started on the assignment, create a new folder in your local repository using Finder/File Explorer. Please name your folder something obvious, like “HW1”. Course staff will access your repository to review your submissions so make sure it’s easy for them to find your submissions.
- **Step 4: Create/edit your files**
    - Please make sure you create all files for your assignment/lab inside the assignment folder you created.
- **Step 5: Add newly created files to Git tracking**
    - When you create a new file, you need to tell Git to keep track of it. Type the following:

```
git add <file-path>
```

        - <file-pathis the path to the new file relative to the folder/directory you’re currently in. So, if you’re in your top level repository folder, which is called repo, and you want to commit a new file called hello.java that’s in a sub-folder called HW1, you would enter

```
git add HW1/hello.java
```

        - You can also easily add all untracked files at once by typing

```
git add .(including the period)
```

- **Step 6: Commit your files**
    - A commit saves a local snapshot of your file for version tracking. It doesn't overwrite any previous versions, so if something goes wrong, you can always revert back to a previous version. Commit often!
    - In Terminal or equivalent, navigate to your local repository folder. Type the following:

```
git commit -m "Your commit summary-should be short and descriptive"
```

    - **Step 7: Push your commit to the remote repository**
        - You do not need to push every time you commit, but it is good practice to push often. Type:

```
git push
```

- You can check that your commit made it to the server by opening the repository in your browser.
- Login to Khoury GitHub in your browser, and open up your repository. You should see your changes. To make sure, click on the tab that says "X commits" and look for your summary message.