ASM Linter - Installation/Configuration Guide

Introduction and Purpose

In this guide, we have included different installation and configuration scenarios relevant to the installation/migration of our runnable software on your machine. We have included sample text, illustrations, and troubleshooting tips to make the overall process more intuitive.

Version Information

- Current Version: JavaLinter 2.4

References

- Java ASM documentation: https://asm.ow2.io/asm4-guide.pdf
- Gradle documentation: https://docs.gradle.org/current/userguide/userguide.html

PreInstallation Requirements

Hardware Minimum Prerequisites

- Processor: Core i3, Core i5, Core i7,
- Processor speed: 1.2 GHz
- RAM: 4GB or more

Software Prerequisites

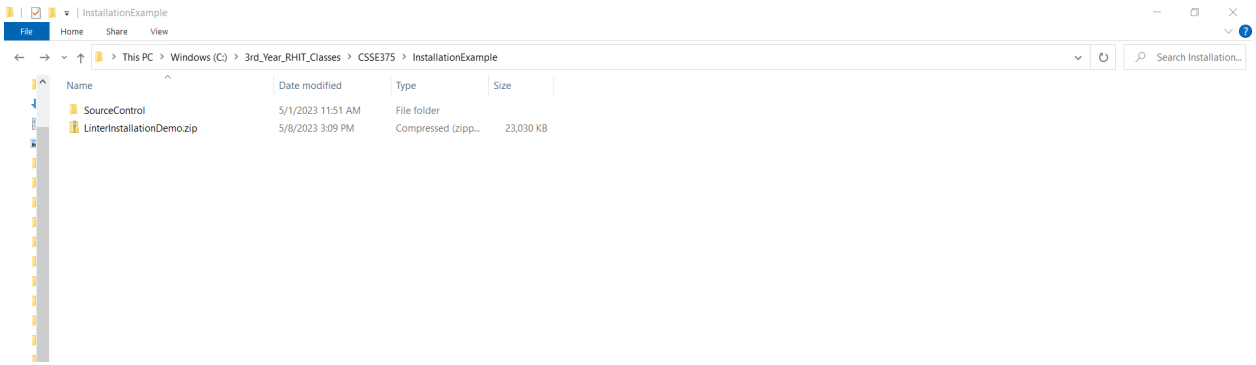
- *Recommended* – Microsoft Windows Server 2003
- Java version: 1.8
- JUnit version: 5.0
- Gradle version: 6.8
- Other libraries our project depends on: ASM 9.2, PlantUML-8059
- An IDE to run your program
    - *Recommended* – Intellij or Eclipse

Developer Installation and Configuration Guide

Instructions if installing from a .zip file

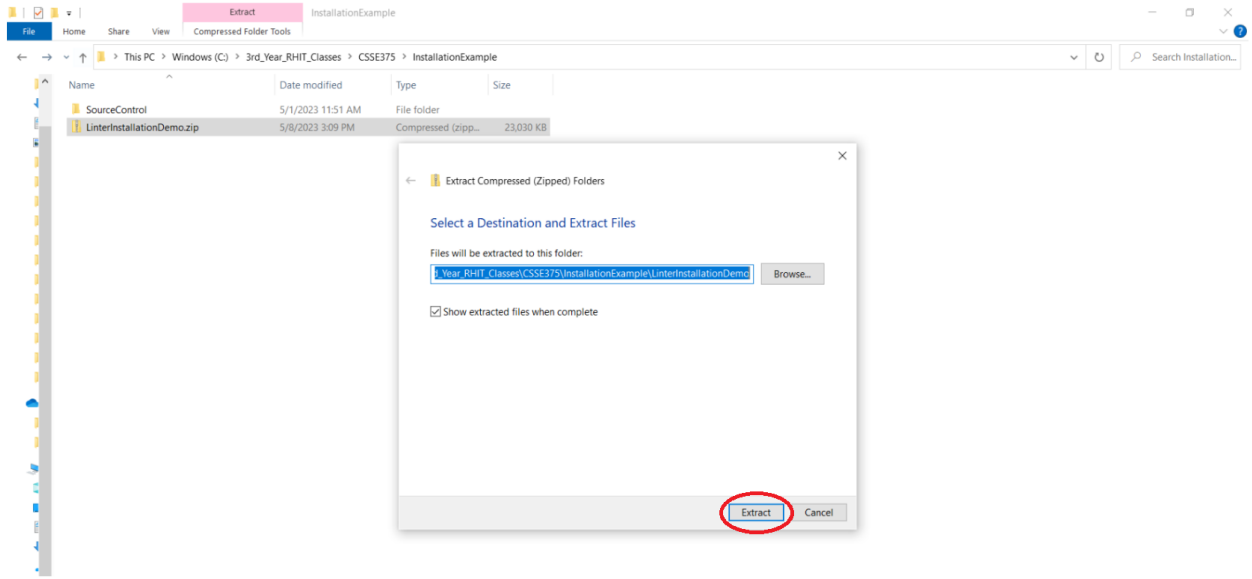
Step 1

Download the software package .zip to your machine. Open the file's location in file explorer.



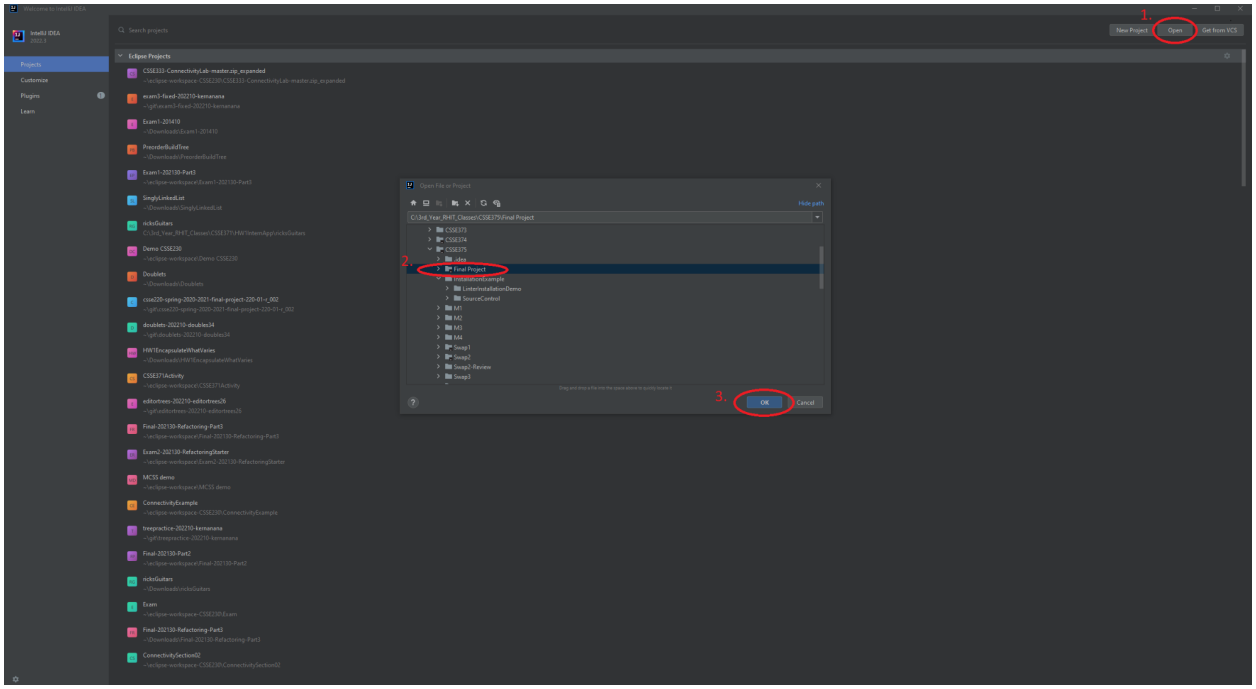
Step 2

Unzip and extract the package at a file location that you can access easily using a method of your choice. The state of the figure below was achieved by right-clicking the zip, then selecting 'Extract All…' -> Extract



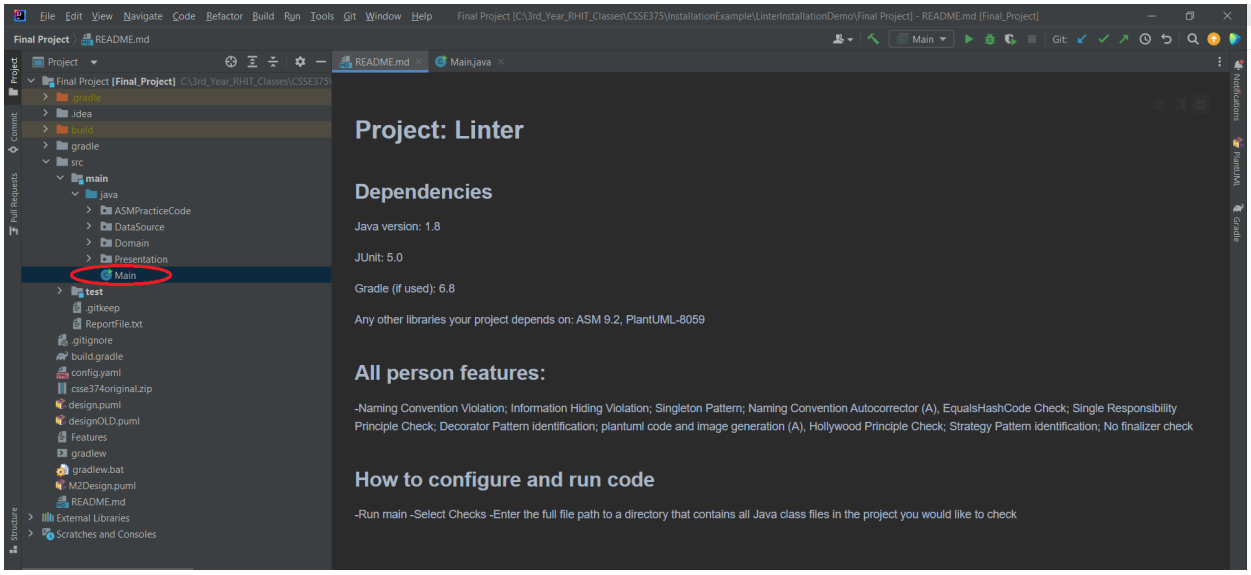
Step 3

From the home menu of your IDE, click the option to open an existing project and then select the unzipped Linter File. Though this step will vary depending on your IDE and its version, most IDE's will allow the option to open a project from an existing file.



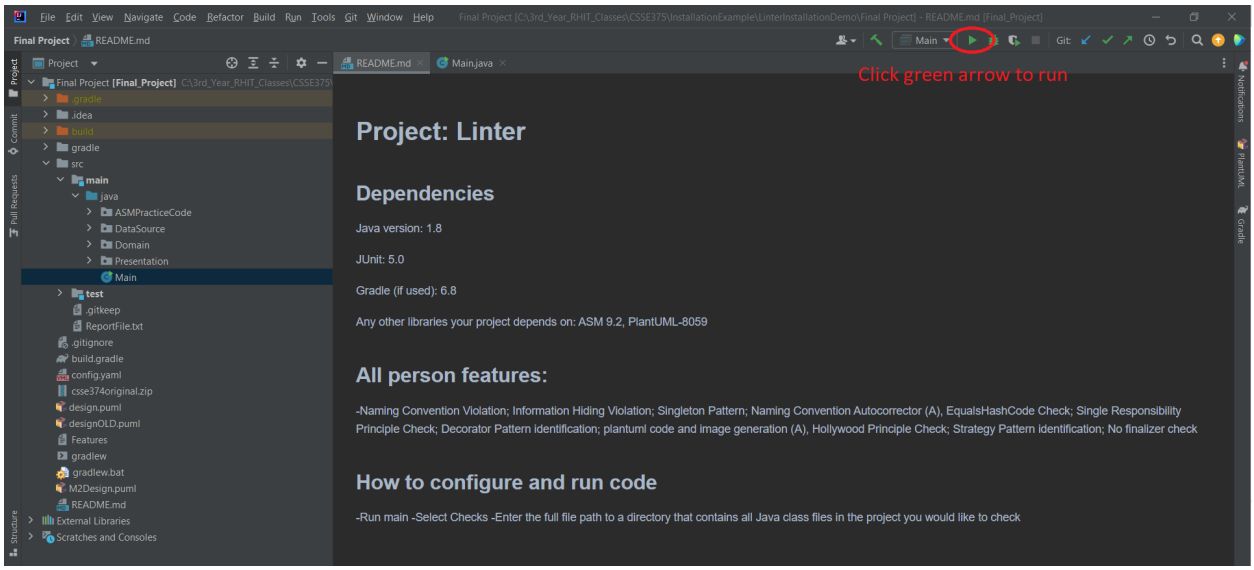
Step 4

Wait for the Gradle Tasks to finish loading. Then locate Final Project -> src -> main -> Main.java.



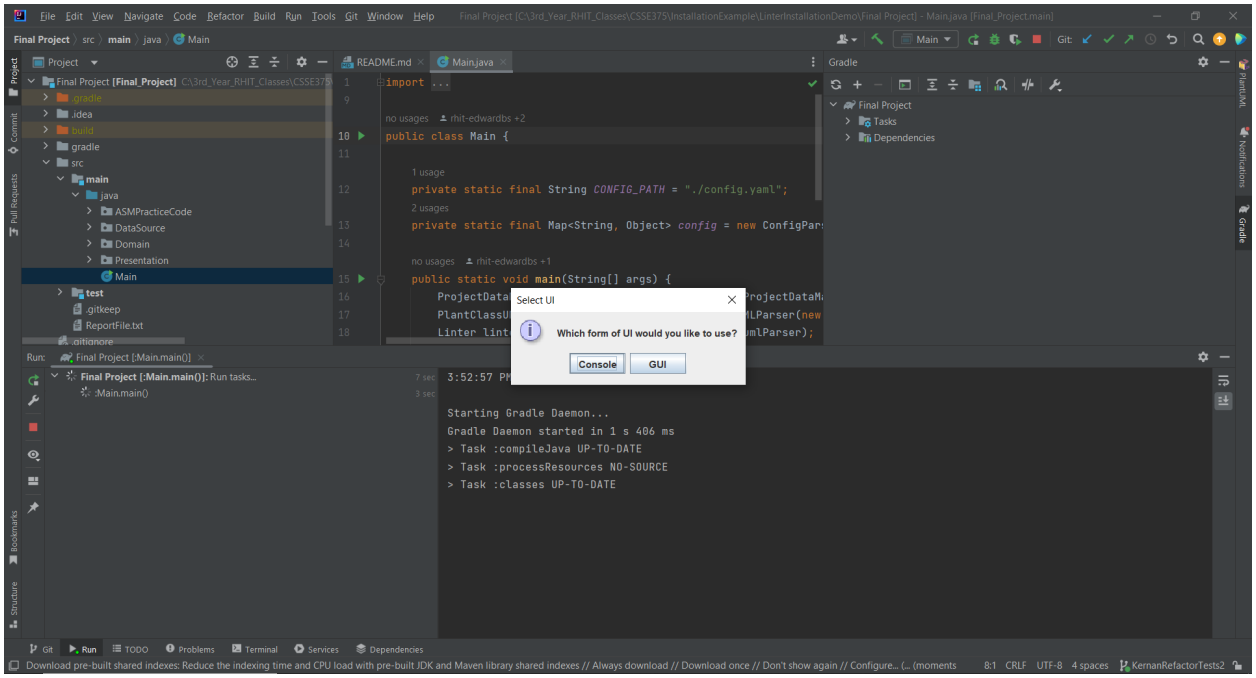
Step 5

Select the main program (double-click in Intellij) and verify that it compiles (refer to troubleshooting if compilation fails). If the console does not print error messages during execution, then installation is completed. Please refer to the User Guide to learn more about using the program itself.



Step 6

Your program should look like the figure below if run correctly.



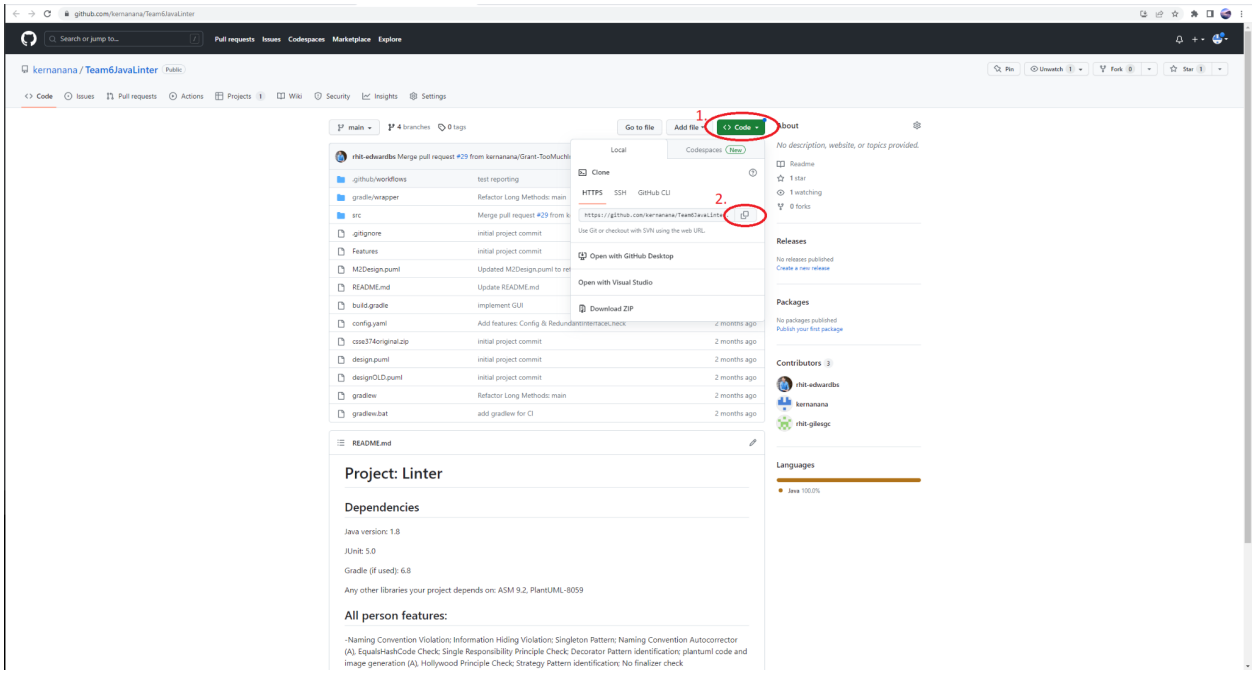
Instructions if importing from Git

Step 1

Visit our code repository on GitHub: https://github.com/kernanana/Team6JavaLinter

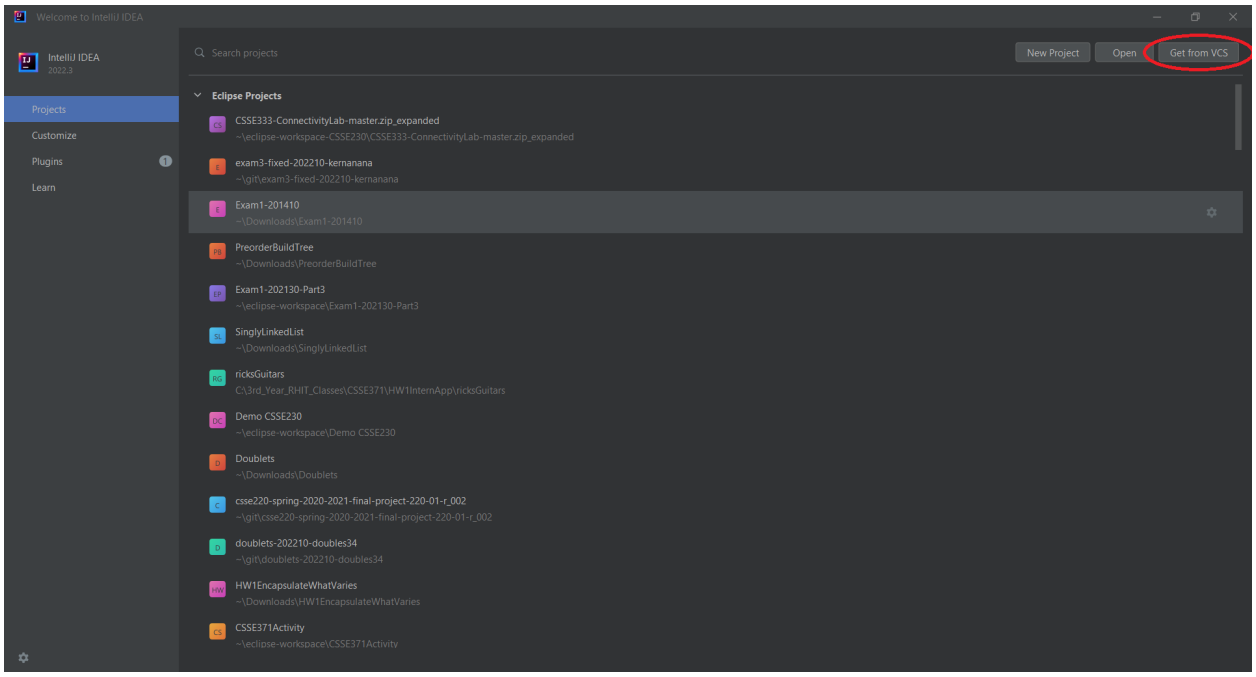
Step 2

On the page, click the green button that reads “Code”, then the copy to clipboard button



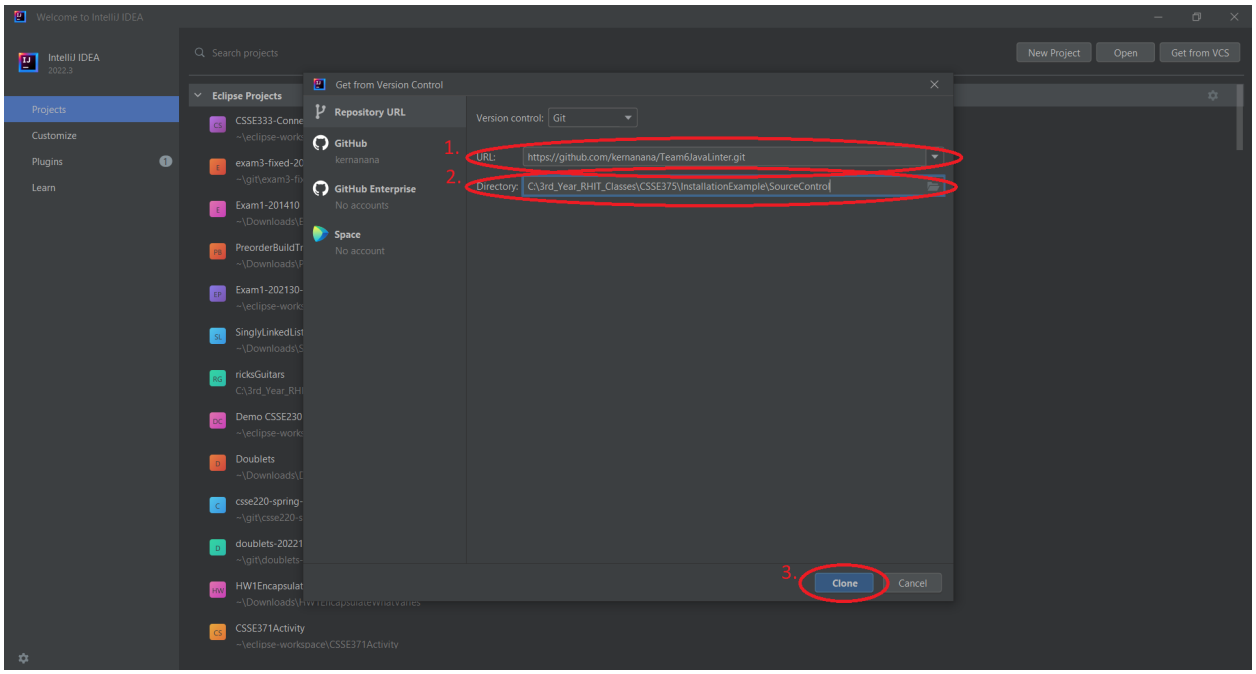
Step 3

Open your Java IDE (we will use IntelliJ Community Edition to demonstrate). Select the option to open a project from version control. From the home screen of Intellij, it will be the ‘Get from VCS’ option.



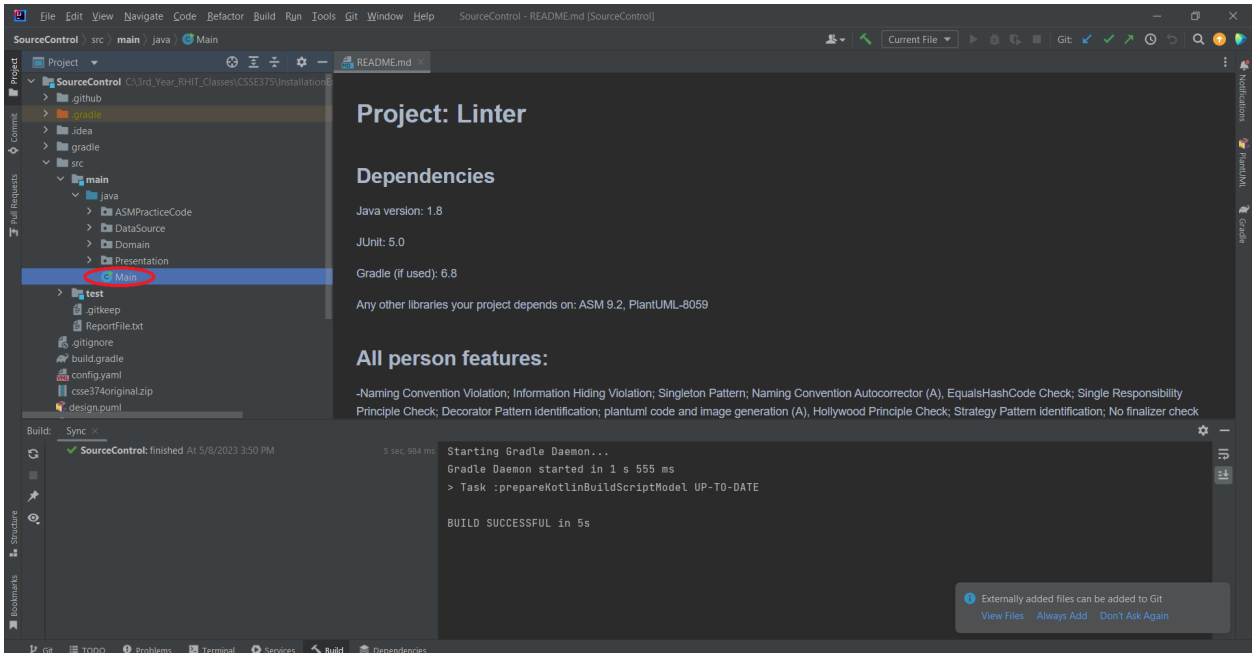
Step 4

Once your IDE's open from version control is selected, paste the link from step2 into the 'URL' textbox. Then select a directory to save the Linter in before clicking 'Clone'.



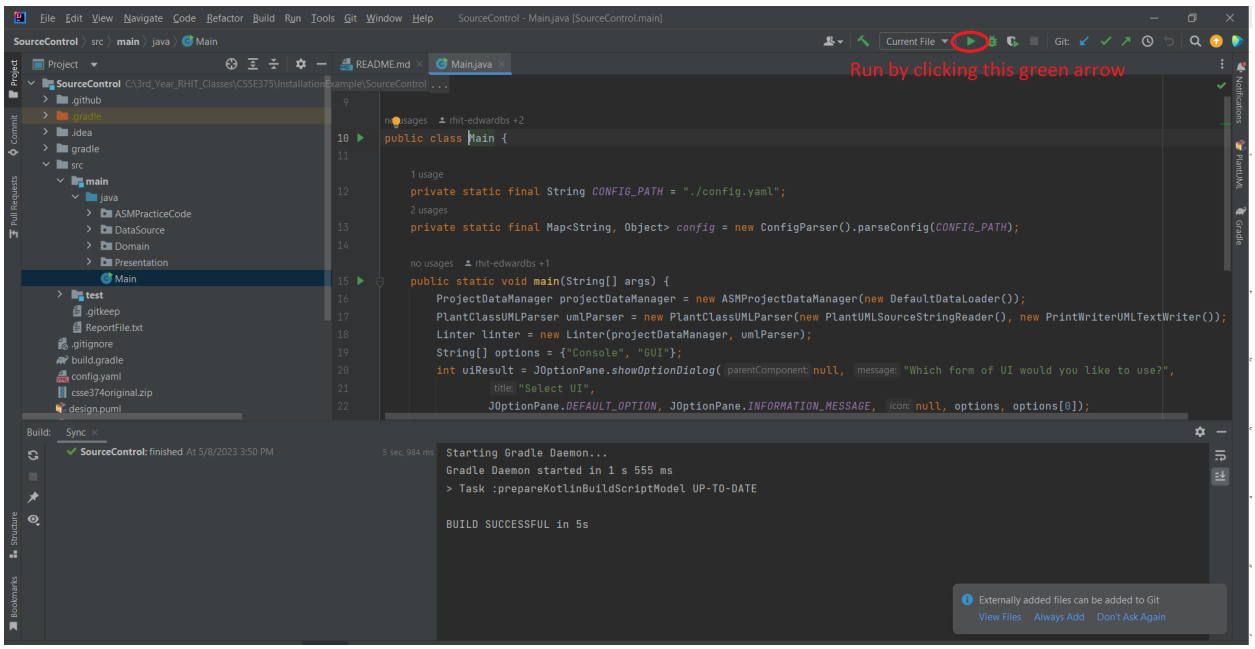
Step 5

Wait for the initial Gradle Tasks to finish executing. Then locate Final Project -> src -> main -> Main.java.



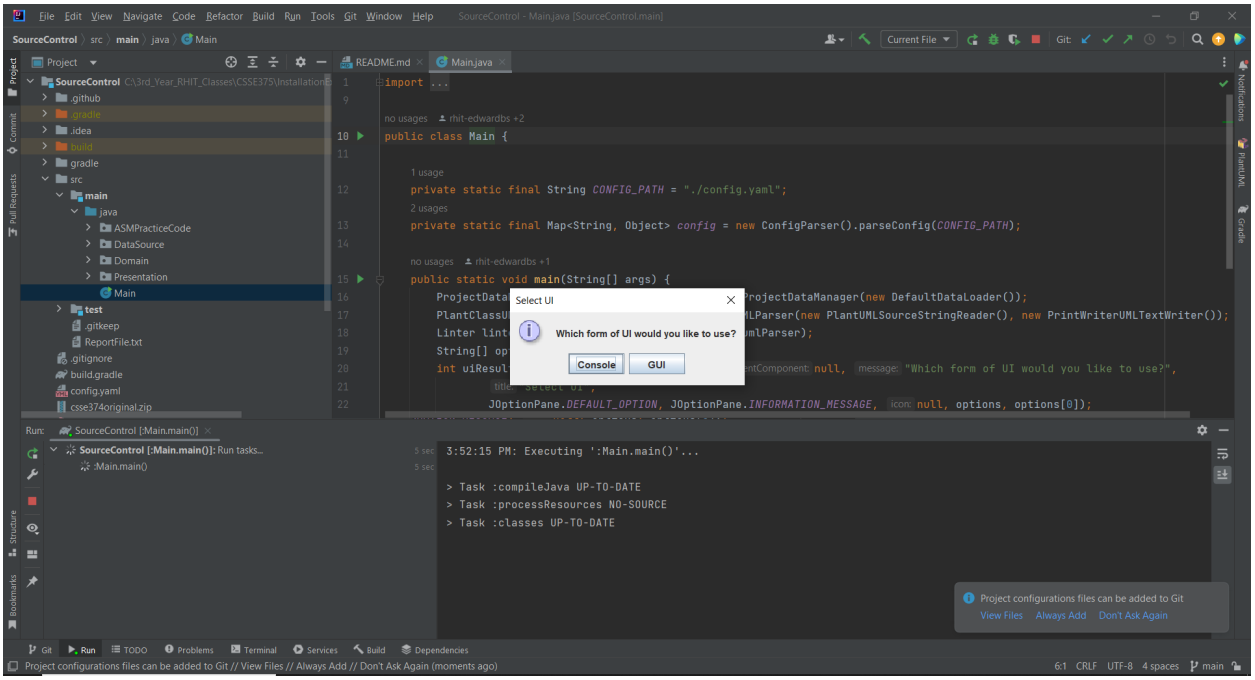
Step 6

Select the main program (double-click in Intellij) and click the run button to verify that it compiles and that installation is set up correctly (refer to troubleshooting if compilation fails). If the console does not print error messages during execution, then installation is completed. Please refer to the User Guide to learn more about using the program itself.



Step 7

Your program should look like the figure below if run correctly.



Troubleshooting

Issue 1: Compilation errors from Java classes are occurring

Possible Cause 1:

The version of the software package you are using has been deprecated.

Solution 1:

Download the latest version: verify that you have downloaded the latest version of the .zip or the link from Github

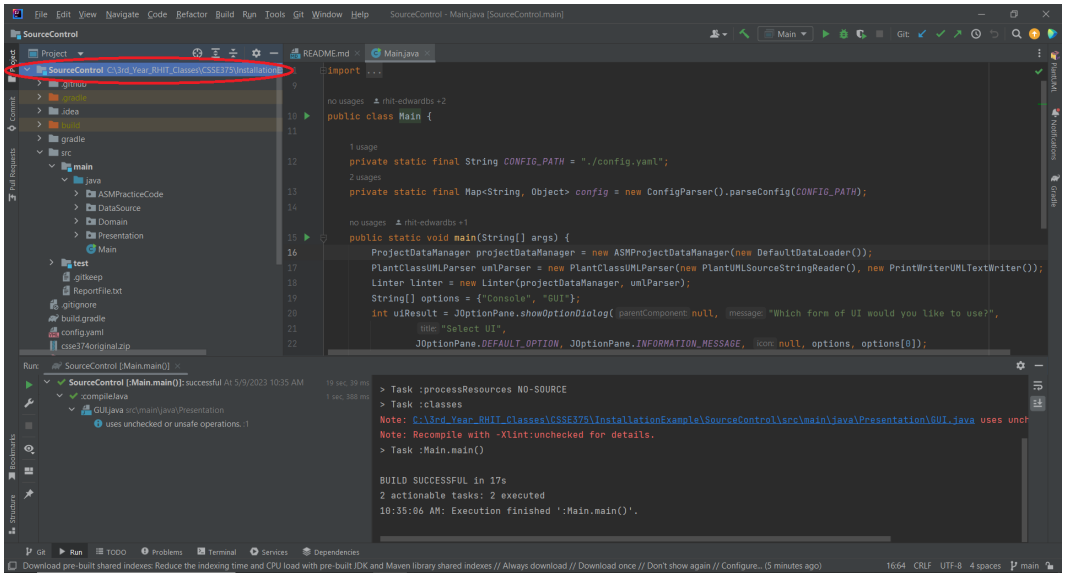
Possible Cause 2:

Check system requirements: it is most likely that the incorrect version of your JDK is being used.

Solution:

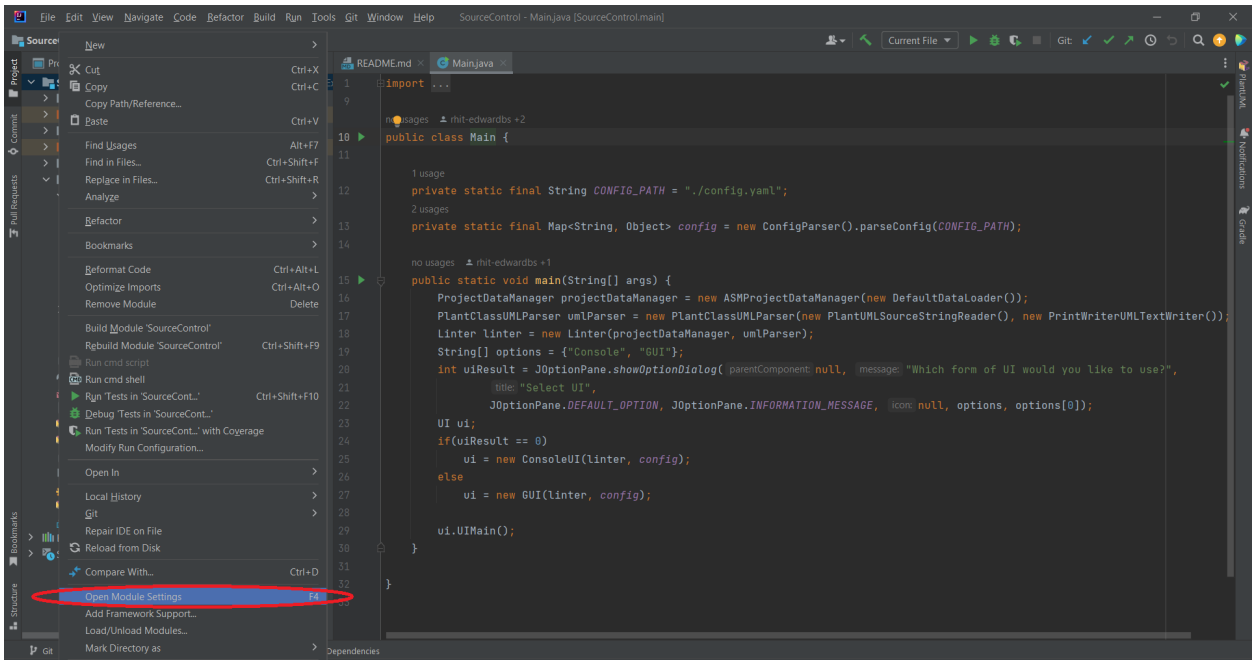
Step 1:

Locate the file location that holds the source code for your linter.



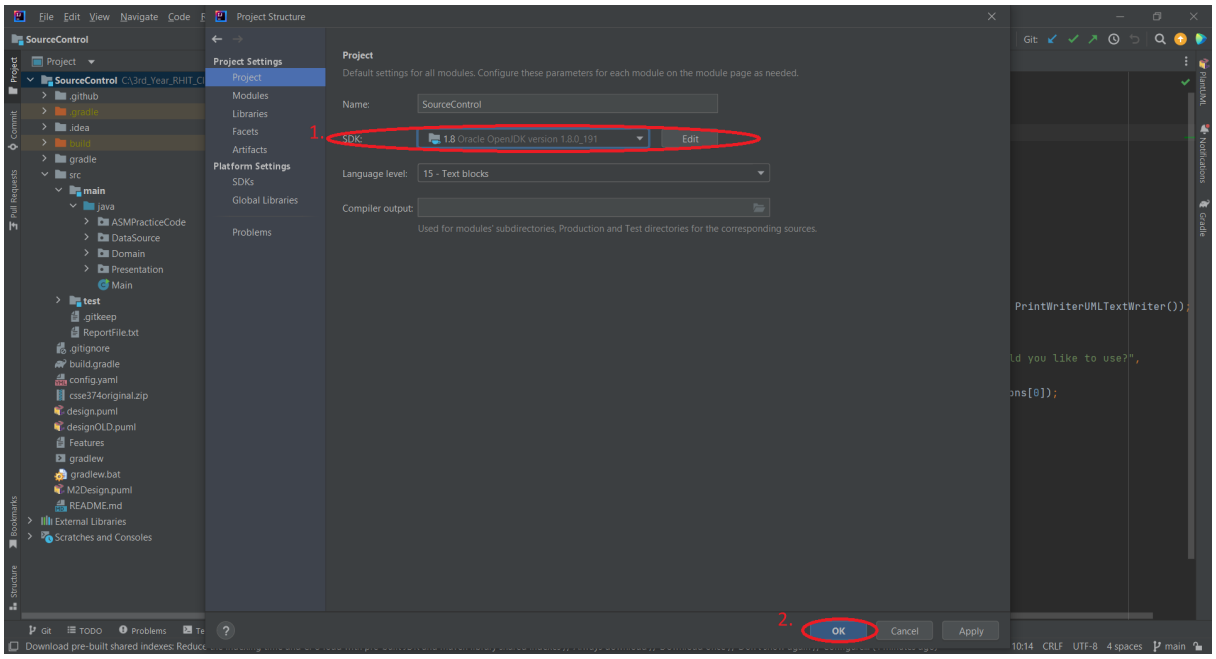
Step 2:

Select the option to open the project's 'Module Settings.' In Intellij, you can right click the project's source folder then click 'Module Settings.'



Step 3

Change the JDK responsible for compiling the project to 1.8, then save your changes. In Intellij, select Project to get to the figure below.



Issue 2: Initial Gradle Tasks are not completing/are not building properly.

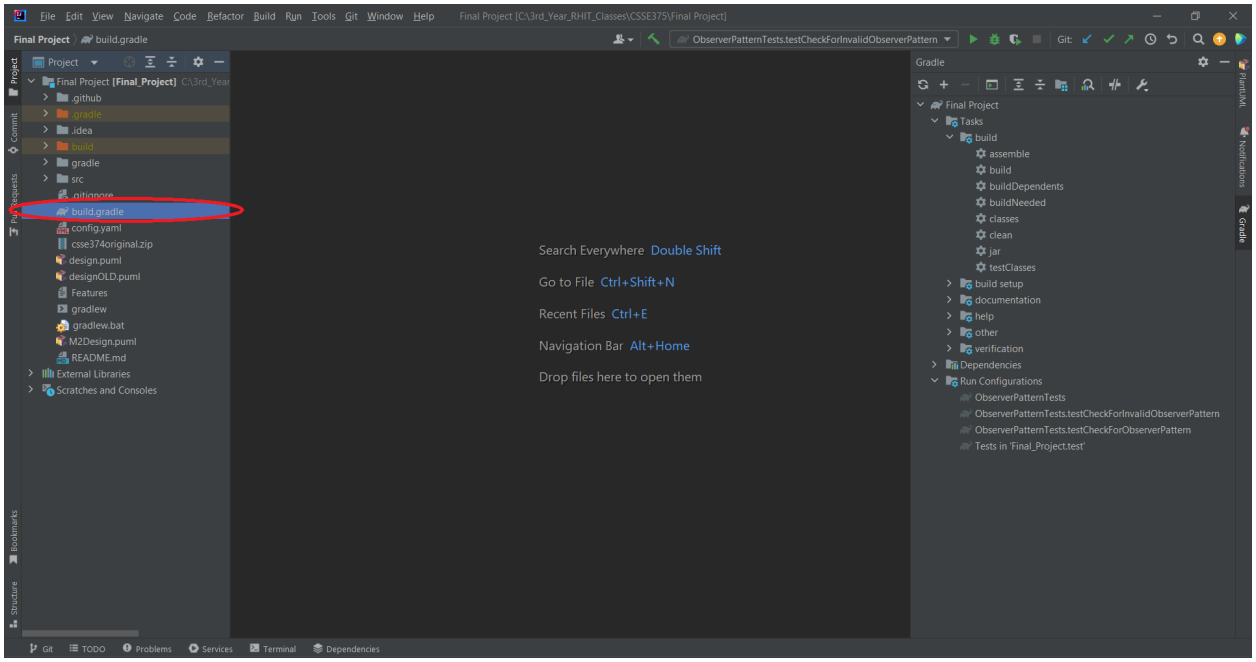
Possible Cause

Gradle dependencies are not set up correctly

Solution

Step 1

In your IDE, find the 'build.gradle' file that is under the root directory



Step 2

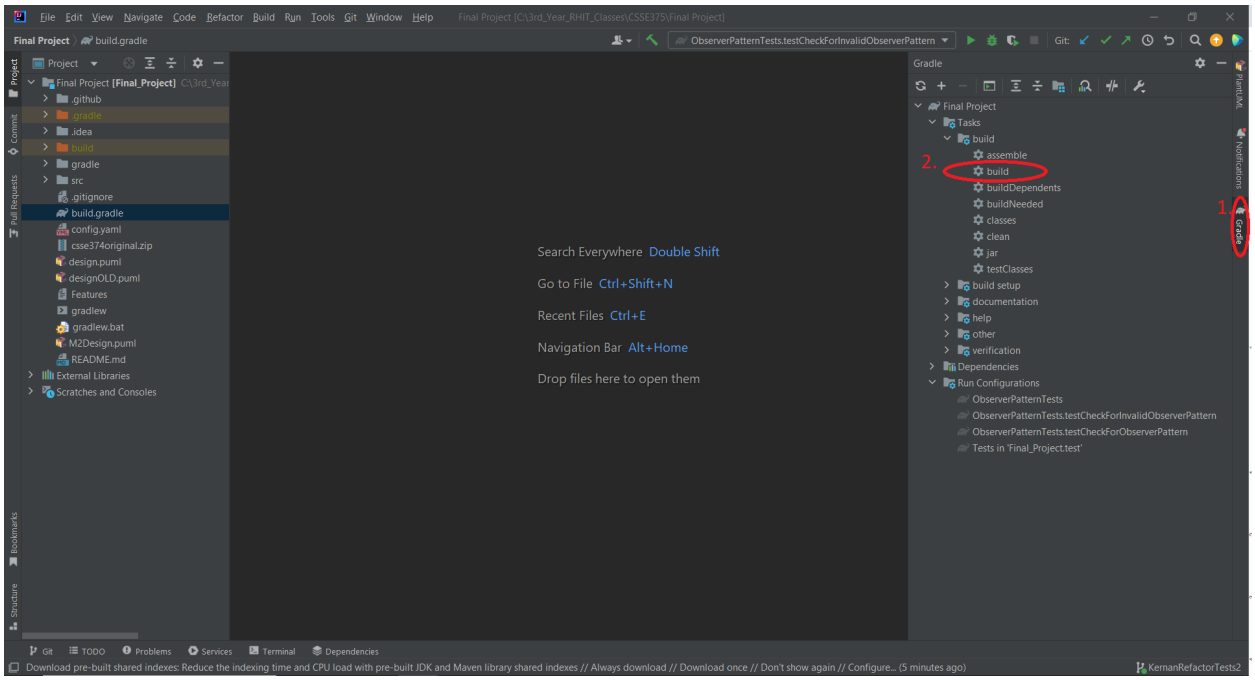
Check to make sure your Gradle script matches ours in the figure below. Copy and paste into the .gradle file if need be.

```
plugins {
    id 'java'
}

group 'org.example'
version '1.0-SNAPSHOT'

repositories {
    mavenCentral()
}

if (hasProperty('buildScan')) {
    buildScan {
        termsOfServiceUrl = 'https://gradle.com/terms-of-service'
        termsOfServiceAgree = 'yes'
    }
}

dependencies {
    implementation group: 'net.sourceforge.plantuml', name: 'plantuml',
version: '8059'
```

```
    implementation group: 'org.yaml', name: 'snakeyaml', version: '2.0'

    implementation 'org.ow2.asm:asm:9.2',
            'org.ow2.asm:asm-tree:9.2',
            'org.ow2.asm:asm-analysis:9.2',
            'org.ow2.asm:asm-commons:9.2'

    testImplementation 'org.junit.jupiter:junit-jupiter-api:5.8.1'
    testRuntimeOnly 'org.junit.jupiter:junit-jupiter-engine:5.8.1'
}

test {
    useJUnitPlatform()
}
```

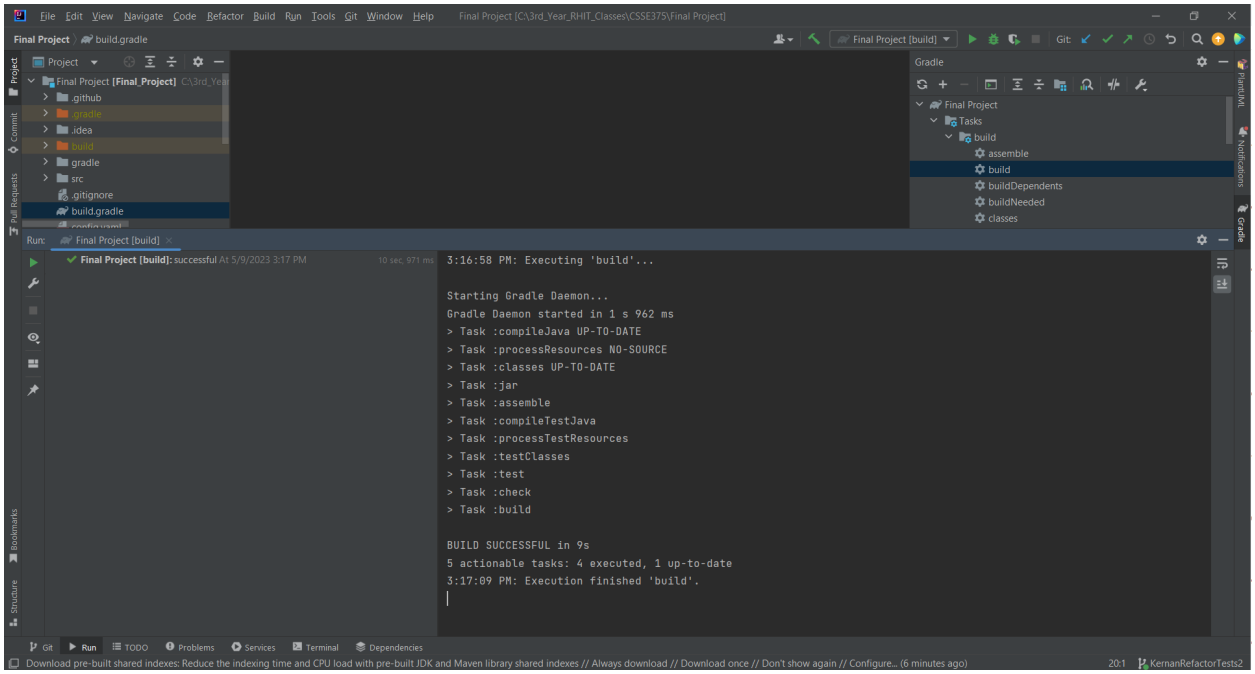
Step 3

In your IDE find the option to reload Gradle projects, then build the project. In Intellij, this is done in the right hand ‘Gradle’ tab as shown below.



Step 4

Verify that the Gradle scripts have run successfully by the console output as shown in the figure below.



Additional Tips

- Verify that the minimum system requirements are met for your device
- Reinstall the program following the steps closely or try an alternate installation method