Framework concepts

1. Create new separate project and build. Use that JAR file in selenium/automation project file.

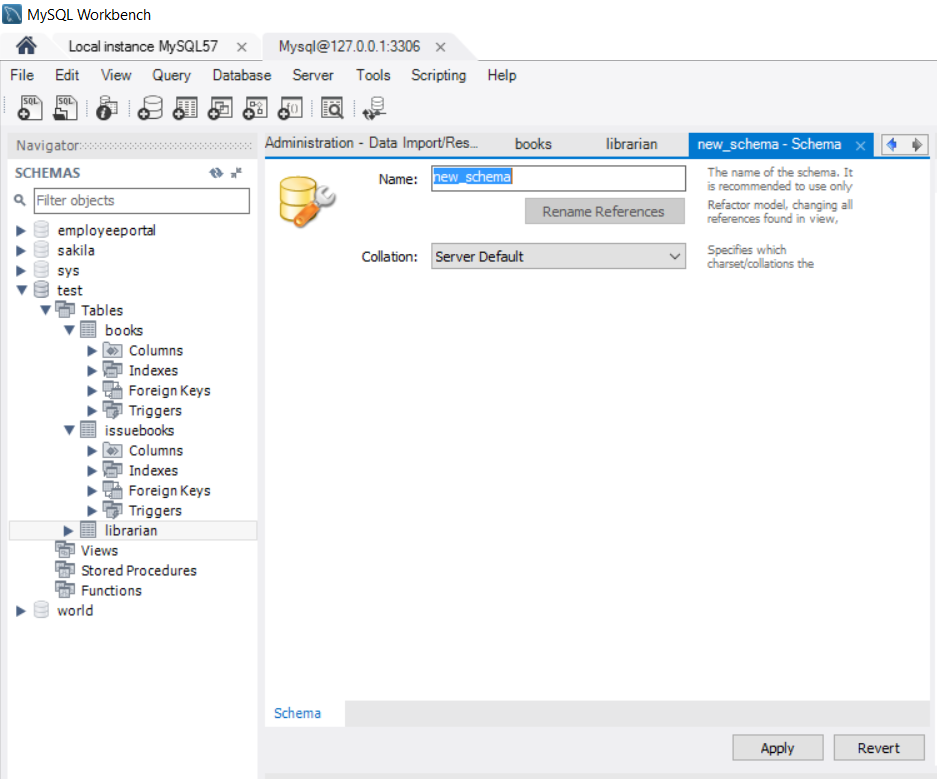
**Creating Java Project and set up database:**

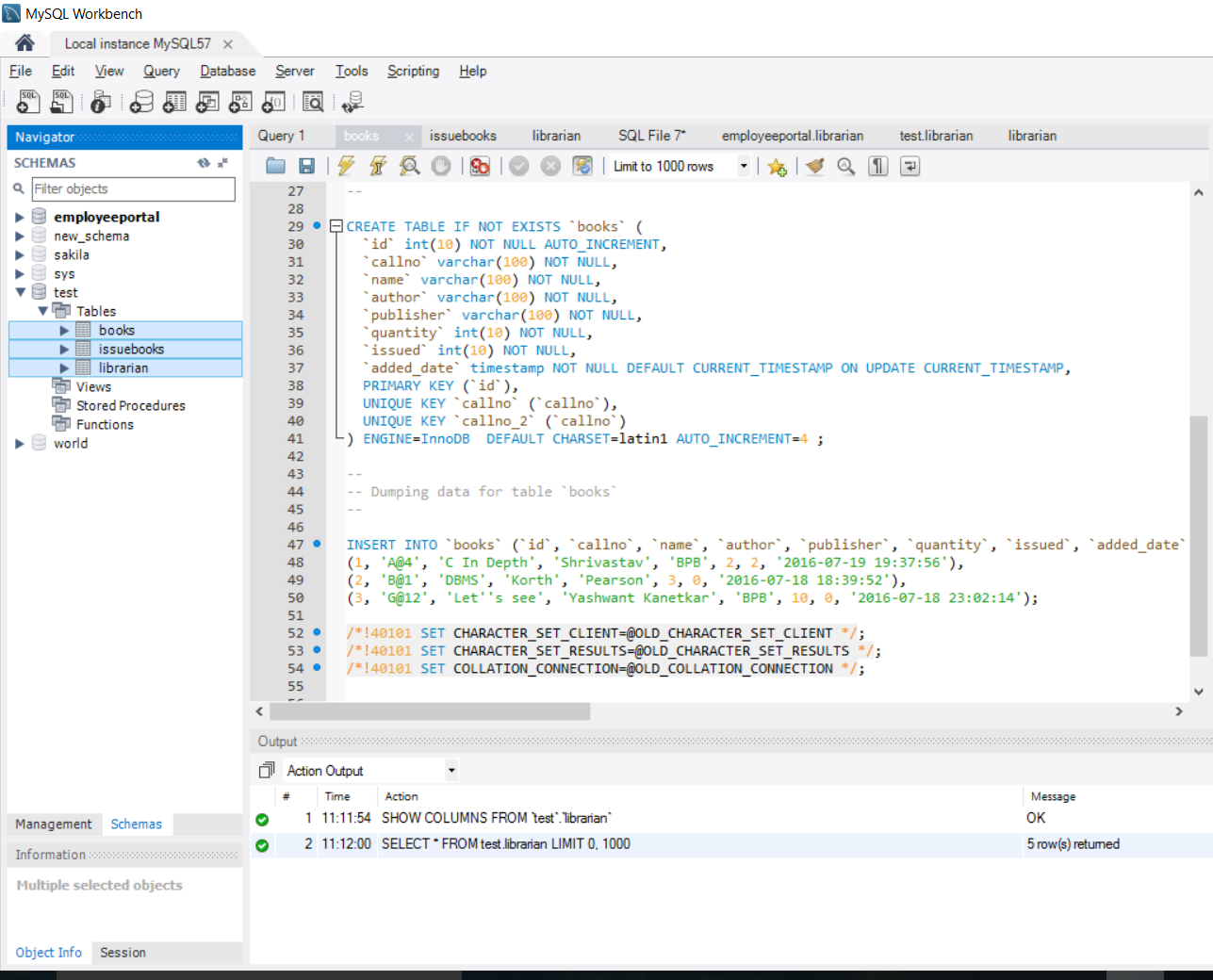
**Step-1: Download java project and import in Eclipse IDE**

* Click file and select open Projects from file systems -> click on directory -> select the project.
* Now download jdbc driver from <https://dev.mysql.com/downloads/connector/j/> and import in the lib folder of the project i.e.., right click on project -> click on properties -> select Java Build Path -> go to libraries -> add external jar files and import jar file.
* Now for giving connection for the jdbc driver give username and password as root and root. Database is connected successfully.

**Step-2: Import .sql files into mySql database**

* At first, create database in mysql file for that select 4th icon which is create new schema in the connected server.
* For importing .sql files we have to go to the server and select Data import -> select Import from Self-contained file and browse .sql file from computer -> set target destination (choose database which is given in the java file) for Default target schema -> start import -> tables will be imported into the database.





* Now, tables are created successfully in the database “**test**”.

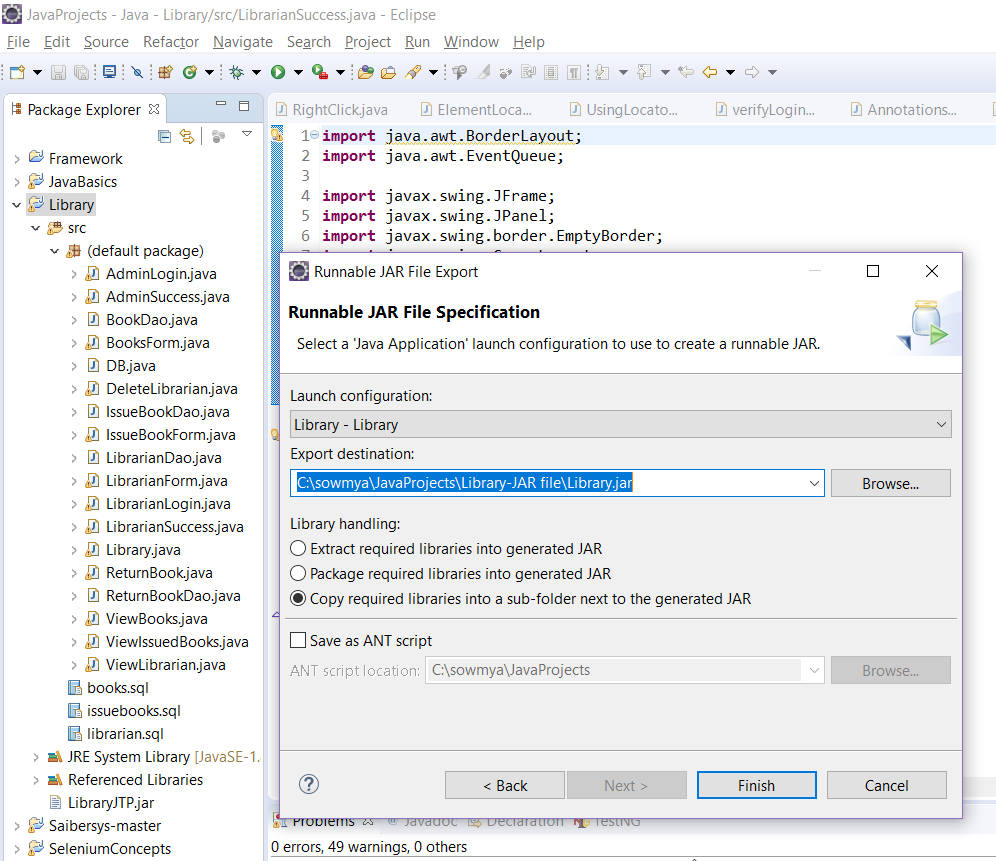
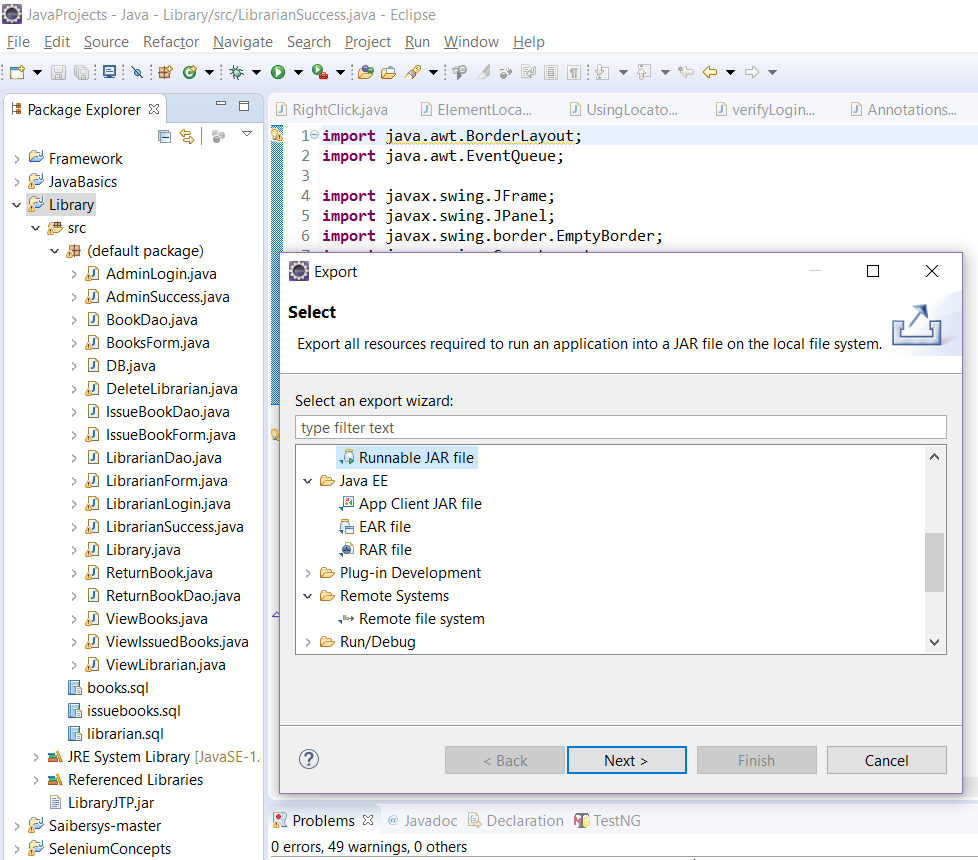
**Step-3: Converting project into jar file**

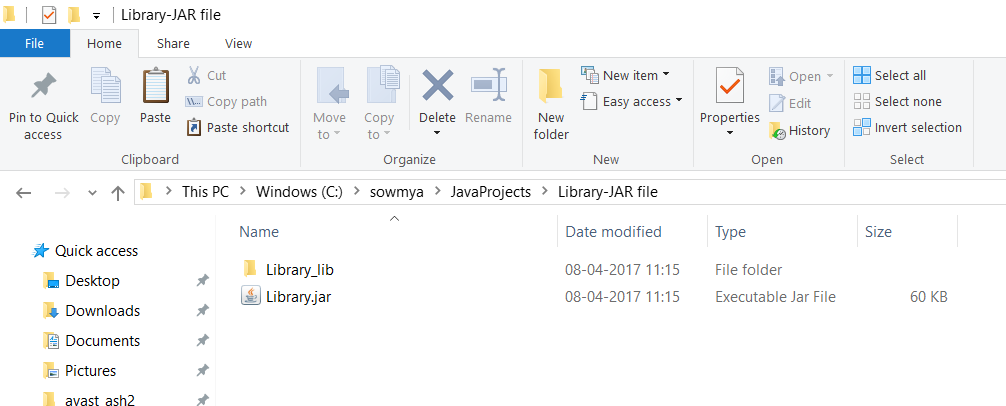
**JAR file:**

JAR (Java Archive) is a package file format typically used to aggregate many Java class files and associated metadata and resources (text, images, etc.) into one file for distribution.

* Select the project and right click on it -> go to export and double click -> Now choose Runnable JAR file from the options and click next.
* Now a dialogue box will appear -> In that select your project in Launch configuration and give the destination path -> click on finish. Finally, the project was turned into a jar file.
* You can view and run the jar file in your system at

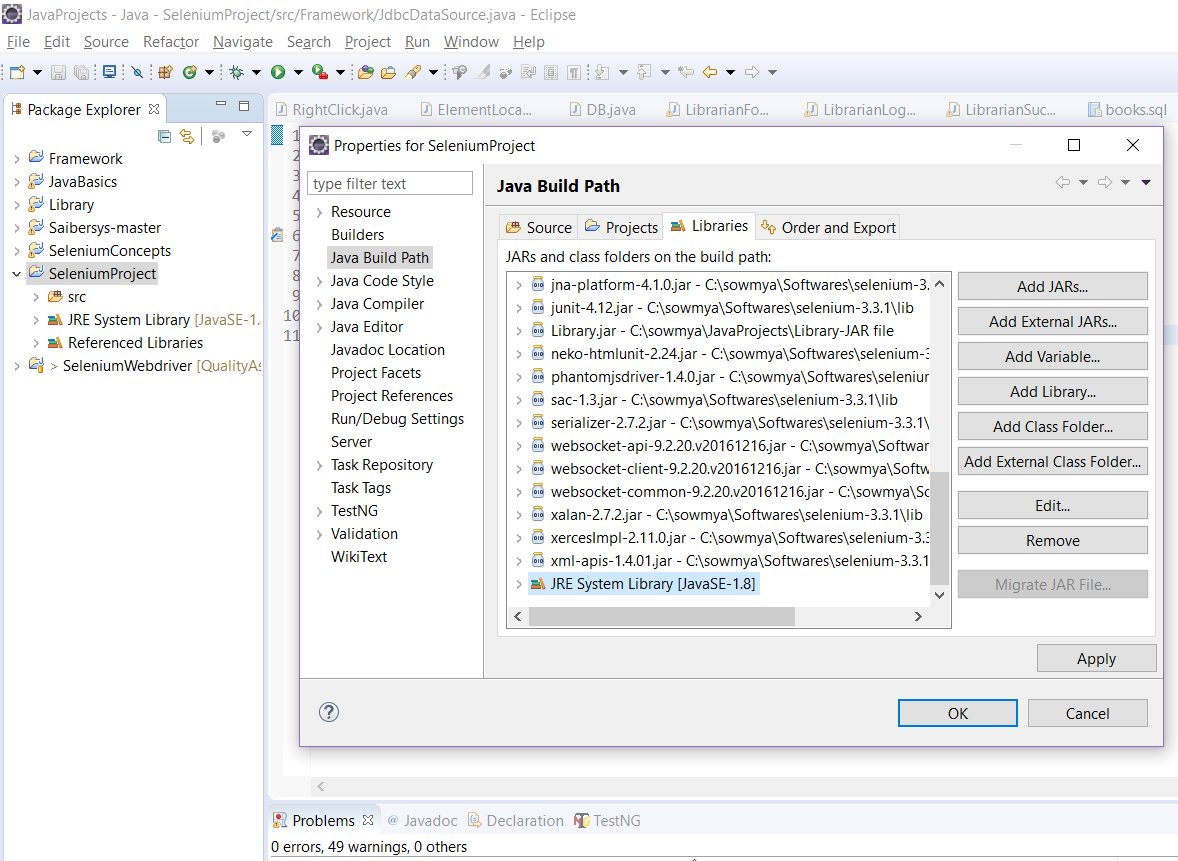
“C:\sowmya\JavaProjects\Library-JAR file”





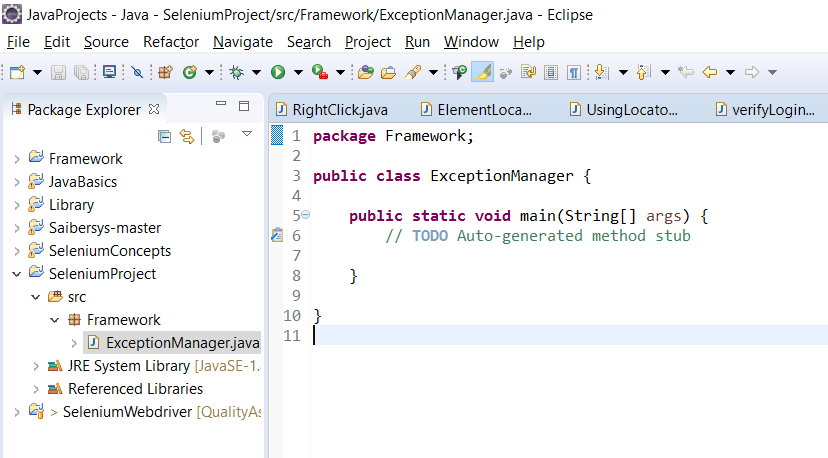
**Step-4: Using above created jar file in Selenium project.**

At first, right click on the selenium project and click on properties -> a dialogue window will appear ->click on java build path and click on libraries -> then click on add external jar file -> Now, add library jar file.

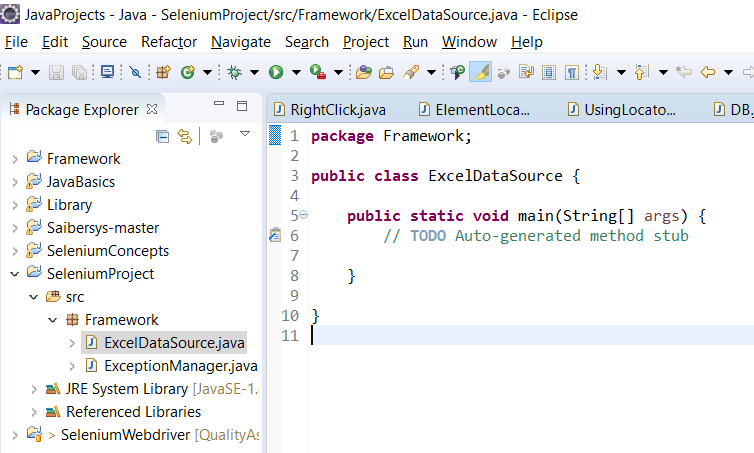


**2)** **Create new class called "Exception Manager" class in framework project under proper package**

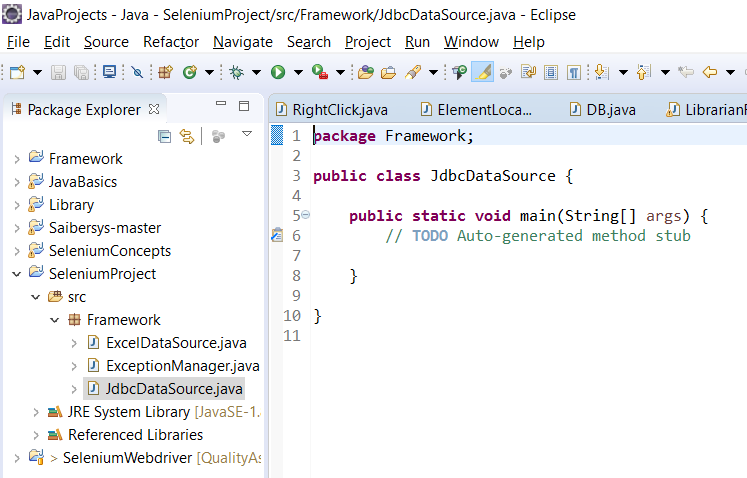
* Initially, create a package as Framework for the project and -> create class **Exception Manager** under that package.



**3) Create new class "ExcelDataSource" for reading data from excel file under proper package**



**4) Create class "JdbcDataSource" for reading data from database table file under proper package**



**5) Create maven project and add selenium jar files in pom.xml**

* At first we need to install maven into our local machine for that go to “ <https://maven.apache.org/download.cgi>” and download the current version, unzip it
* Now, we have set environmental properties and path just like java ->

**Step:1**

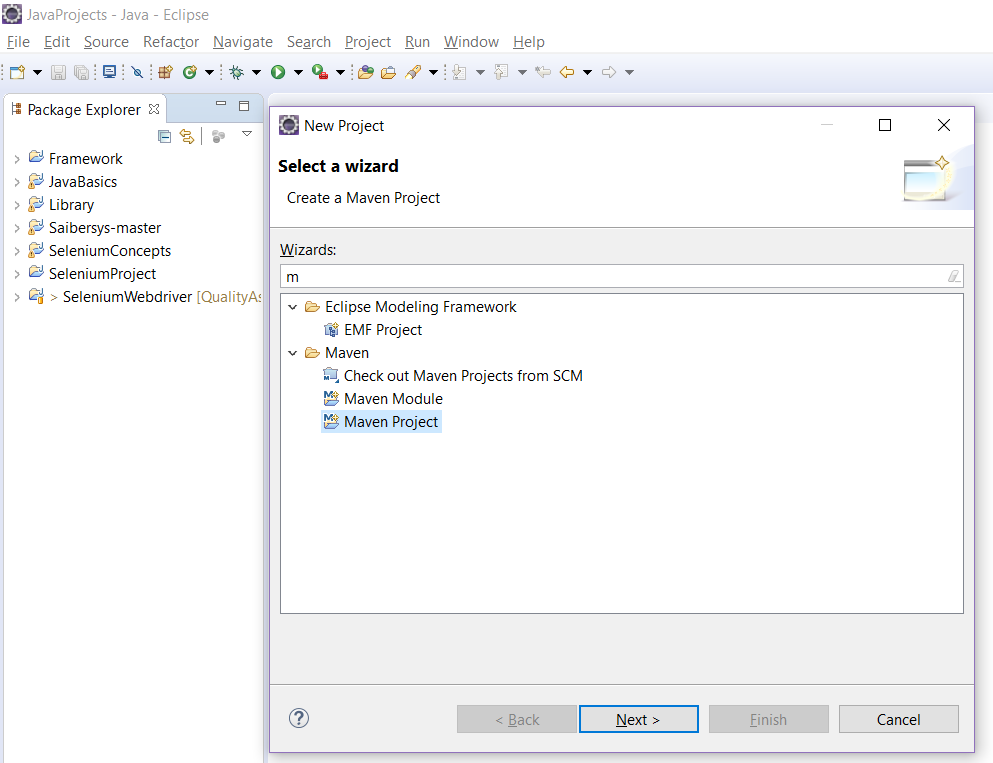
* Go to folder where maven is downloaded and open maven -> copy the entire path -> right click on my computer and click on properties -> click on advance system settings and select environment variables -> select new in system variables, then enter variable name as MAVEN\_HOME and paste the above copied path and click on ok

**Step:2**

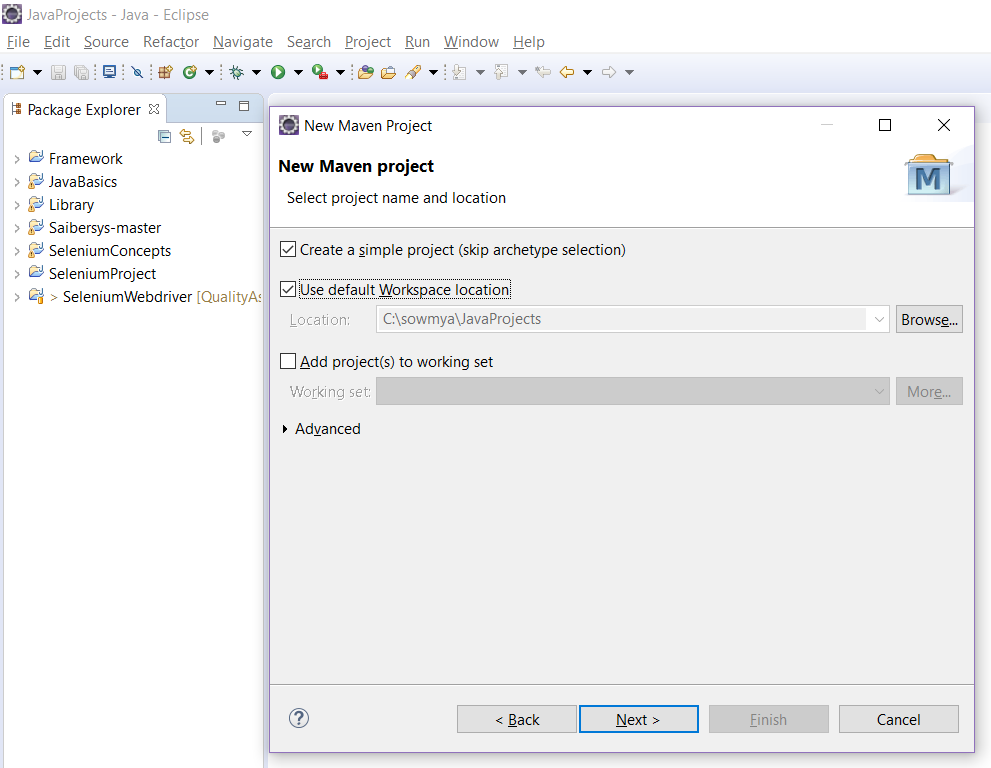
* Go to the folder where maven is downloaded -> open maven and open bin folder and copy the entire path -> right click on my computer and click on properties -> click on advance system settings and select environment variables -> edit the path from system variables by adding “;” at the end and paste the above copied path -> click on ok
* Now, open cmd and type mvn -version -> then if you get the details of maven then it got successfully installed.

Now, Since my Eclipse version is neon I don’t need to integrate selenium with maven.

Initially, go to File and select new -> click on project and choose Maven Project -> click on next



Now, check the 2 options as shown in the figure -> click on next.



**POM (**Project Object Model):

It is fundamental Unit of Work in Maven. It is an XML file. It always resides in the base directory of the project as pom.xml. The POM contains information about the project and various configuration detail used by Maven to build the project(s) like plugins, project dependencies, goals, project version, developers, mailing list and build profiles.

* All POM files require the project element and three mandatory fields: groupId, artifactId, version.
* Projects notation in repository is groupId:artifactId:version.

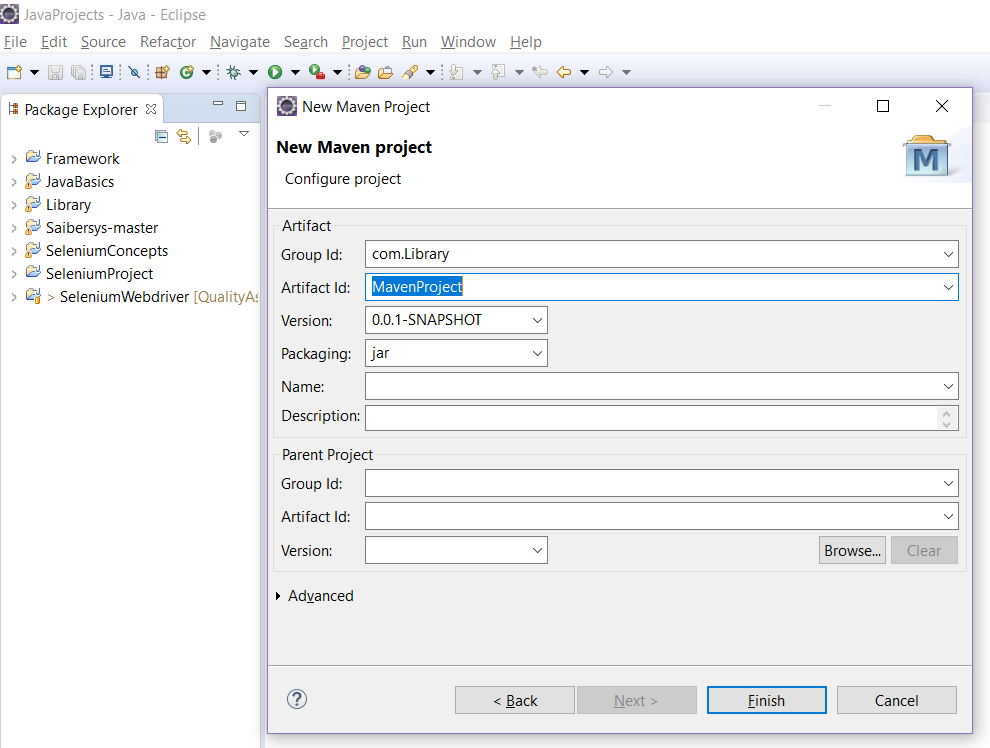
groupId: This is an Id of project's group. This is generally unique amongst an organization or a project. For example, a banking group com.company.bank has all bank related projects.

artifactId: This is an Id of the project.This is generally name of the project. For example, consumer-banking. Along with the groupId, the artifactId defines the artifact's location within the repository.

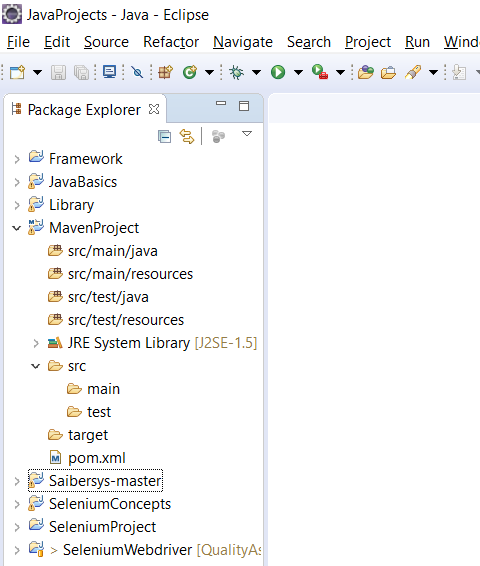
Version: This is the version of the project.Along with the groupId, It is used within an artifact's repository to separate versions from each other. For example:

com.company.bank:consumer-banking:1.0 / com.company.bank:consumer-banking:1.1.

* Now, give groupId, artifactId and version and your Maven project is created



* Maven project is created with pom.xml file



From the above figure, it has four folders :

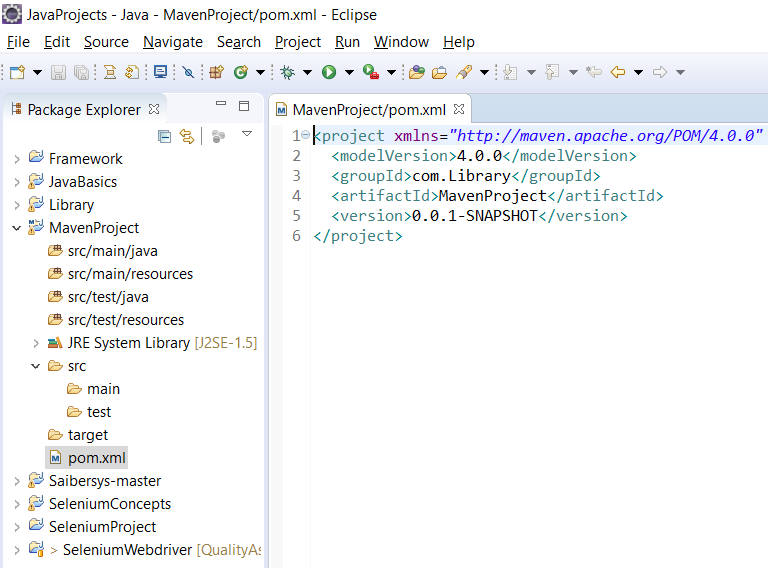
**src/main/java:** It is for source files

**src/main/resources:** It contain resources for the project like xml files, images..etc

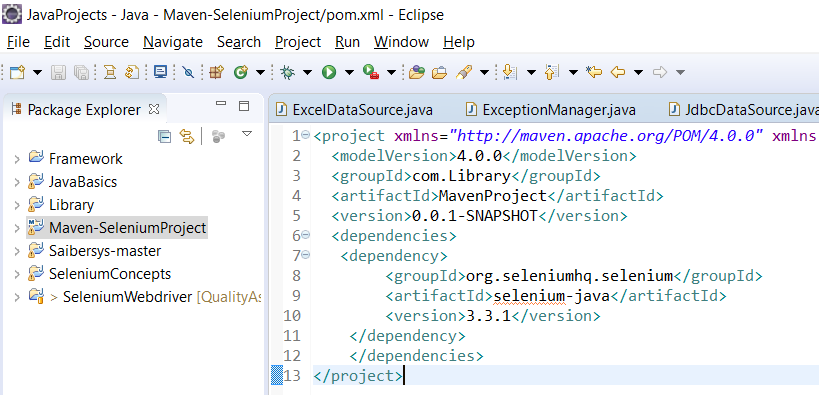
**src/test/java:** It contains test source files

**src/test/resources:** It contains resources of test file.

* Next, in the src we have two directories like main and test, also we have pom.xml file

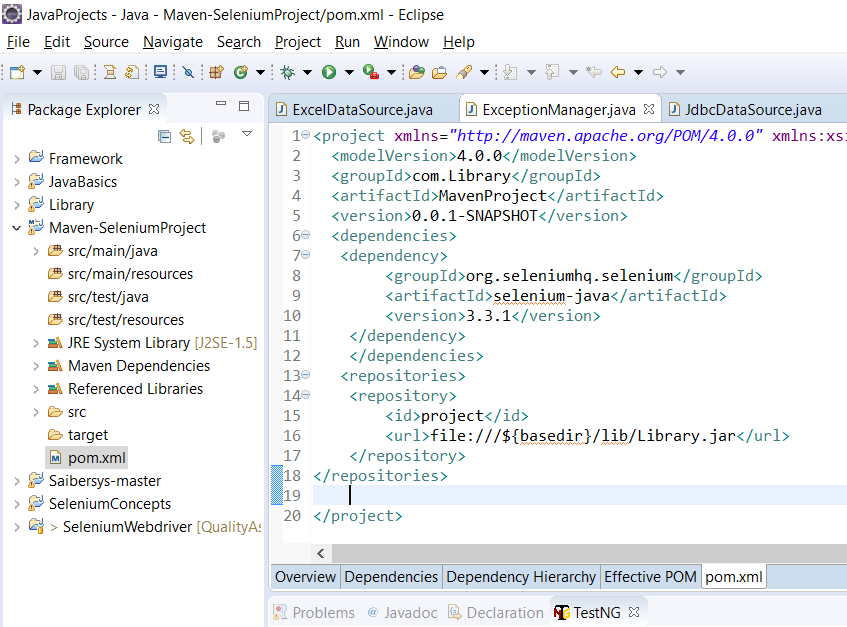


Now, for adding Selenium jar files we have to add its dependencies in the pom.xml file. For that we have to visit <http://docs.seleniumhq.org/download/maven.jsp> and add the dependency.



**6) Add above created jar file also into pom.xml**

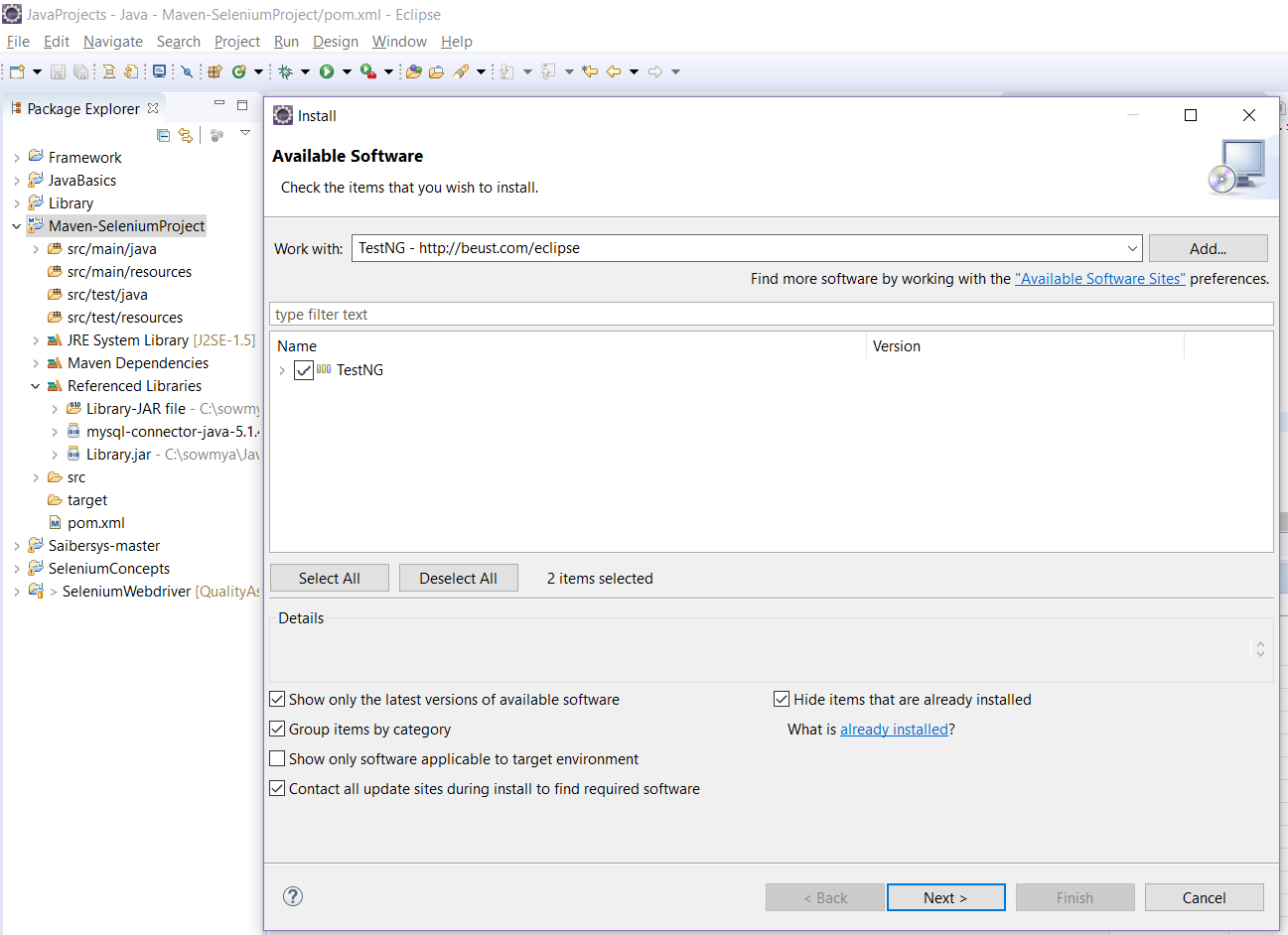
Now, we need to add above created project jar file into it. For that open pom.xml file and add jar



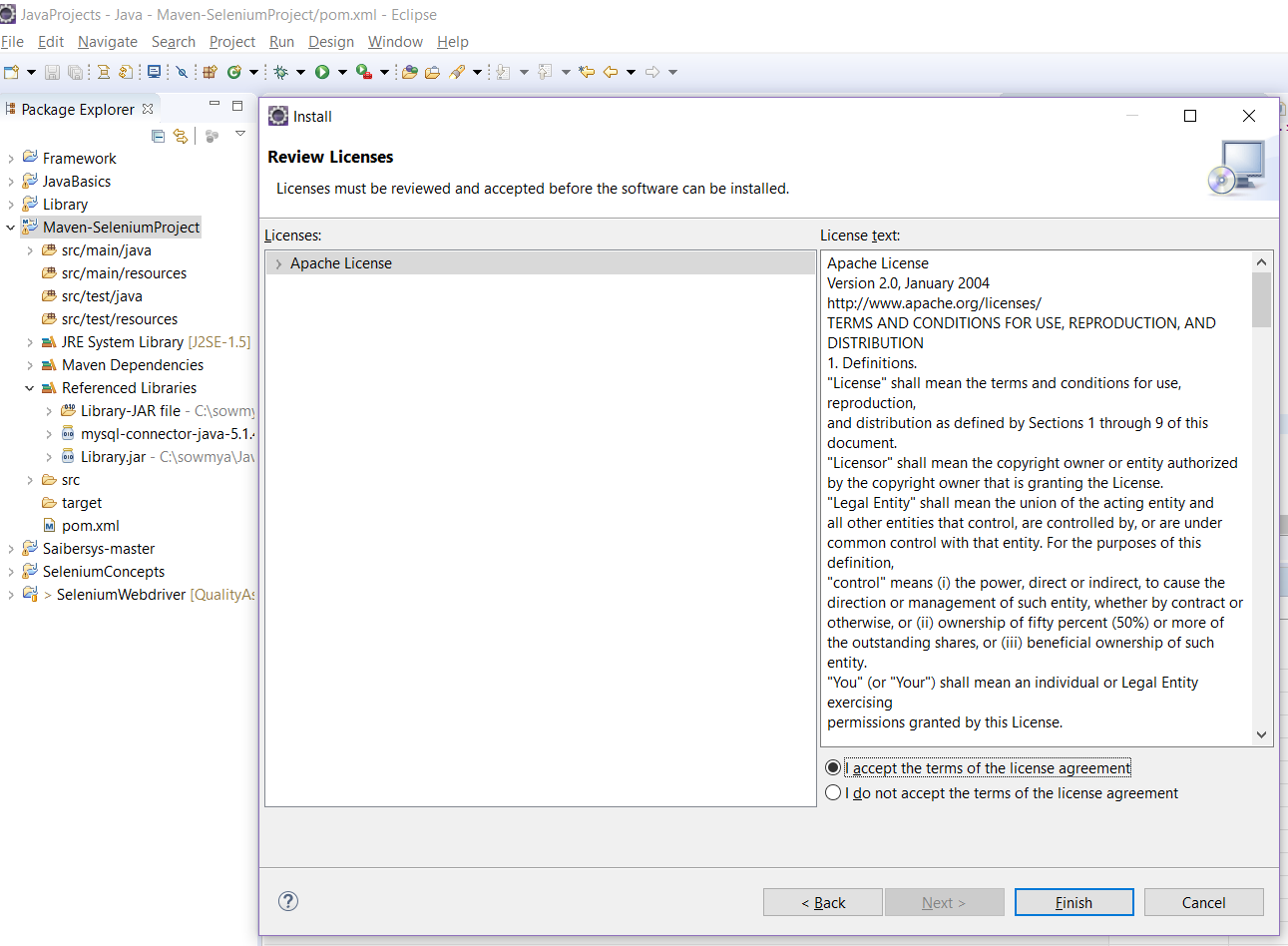
**7) Configure TestNG in eclipse**

Initially go to Help and click on install new software -> type

“TestNG - <http://beust.com/eclipse>” in Work with and press enter, an option TestNG will be populated below -> Now, check it and click on next



After that, check the agreement and click on finish -> Now, once restart eclipse and TestNG got installed



**PAGE OBJECT MODEL:**

It is a **design pattern** which creates/behaves like an object repository for web UI elements (locators) which can be readable and maintained easily and can be reused. It reduces duplication of code.

**Flow:**

* For each web page in the application there should be corresponding page class
* This **Page class (pure java classes)** will find all the locators of that web page and also contains Page methods (name should be given as per the task). which perform operations on those locators. Call them from the test in which you have to use. So the benefit from this will be if any changes in Page then you do not have to modify the test simply modify the respective page and that all.
* So, here you can create a layer between your test script and application page, which you have to automate. That is, it behaves like an **Object Repo** for web UI elements.

**Advantages:**

* Code becomes less and optimized because of the reusable page methods in the POM classes.
* Page Object Patten says operations and flows in the UI (page class) should be separated from verification (test). This concept makes our code cleaner and easy to understand.
* Object Repo is independent of test cases so, we can use the same object repository for a different purpose with different tools. For example, we can integrate POM with TestNG/JUnit for functional Testing and at the same time with JBehave/Cucumber for acceptance testing.
* Methods get more realistic names which can be easily mapped with the operation happening in UI. i.e. if after clicking on the button we land on the home page, the method name will be like 'gotoHomePage()'.

**PageFactory:**

Page Factory is an inbuilt Page Object Model concept for Selenium WebDriver but it is very optimized.

* Here as well, we follow the concept of separation of Page Object Repository and Test Methods. Additionally, with the help of PageFactory class, we use annotations @FindBy to find WebElement. We use initElements method to initialize web elements

**Implementation of Page Object model using Selenium Webdriver:**

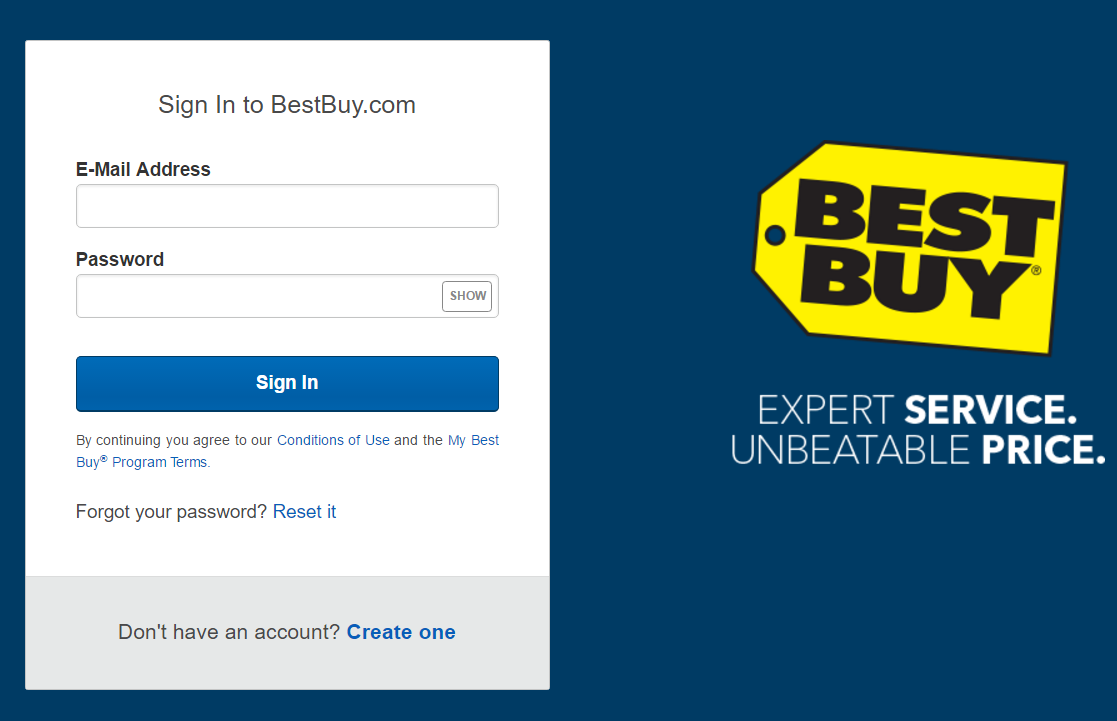
We have two choices and you can use any of it.

1. Page Object model without PageFactory
2. Page Object Model with Pagefactory.

**Page Object model without PageFactory:**

* Let’s take very basic scenario which you can relate to any application. Consider you have login page where Email Address, password, and Sign In button is present.

**Example**: Let’s take Best Buy site



**Step-1:**

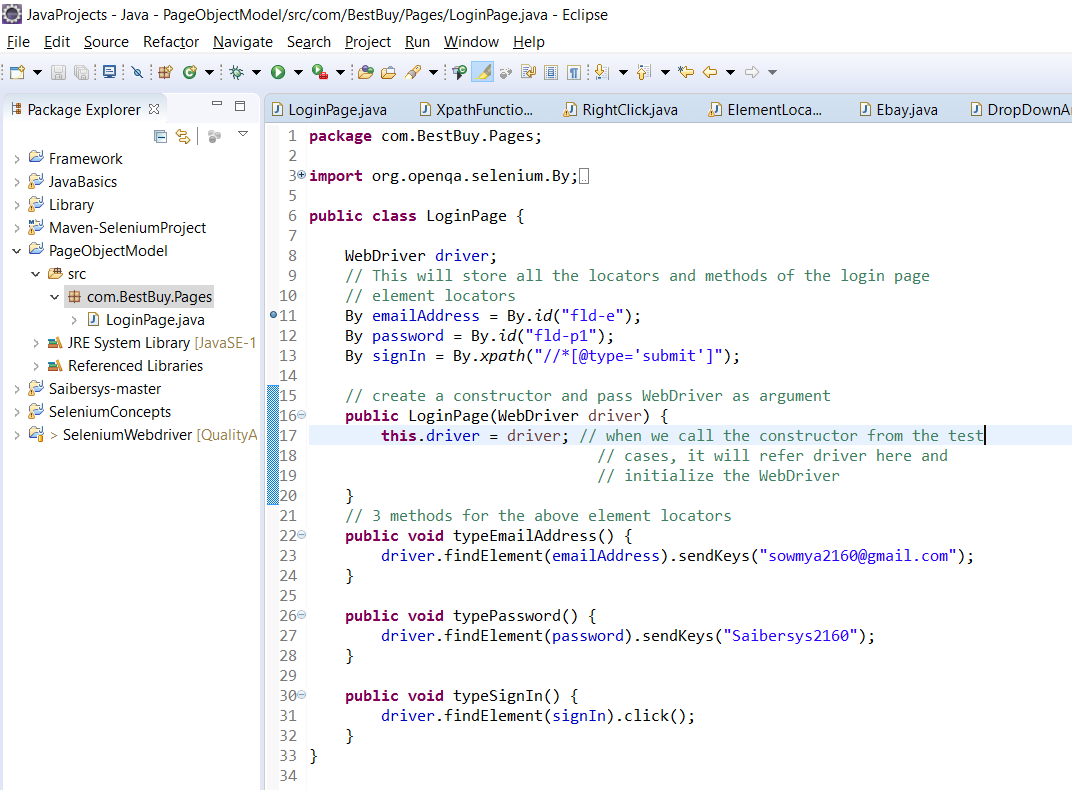
Now, for the above Login page will store three locators – email, password and SignIn and will create methods to access them.

* At first, create a project and create a package com.BestBuy.Pages -> create a LoginPage class inside it and write

3 element locators

1 constructor

3 methods element locators



**Step-2:**

* Now, I want to design test case so I can use the Login class, which I created and can call the methods accordingly.
* For that create another package com.BestBuy.Testcases and create a VerifyBestBuyLogin class.
* Write the steps to invoke the website -> create an object for the java class LoginPage to call all the methods of it.
* Import TestNG to execute the test cases in the order which is given with the help of annotations
* After that execute and you can see the test cases as passed.

