Table of Contents

[User guide 2](#_Toc535844917)

[Setup 2](#_Toc535844918)

[Prerequisites for Setup 2](#_Toc535844919)

[Project Setup 2](#_Toc535844920)

[Project Structure 2](#_Toc535844921)

[Code Structure 2](#_Toc535844922)

[Configuration, test data and logging 4](#_Toc535844923)

[Steps to create runnable jar 4](#_Toc535844924)

[Test Case Execution 5](#_Toc535844925)

[Reports 6](#_Toc535844926)

[Test scenarios covered 6](#_Toc535844927)

# User guide

## Setup

### Prerequisites for Setup

* Install Eclipse (To run the tests from IDE and view the code). Link for steps : <https://www.ntu.edu.sg/home/ehchua/programming/howto/EclipseJava_HowTo.html>
* Install and Setup JAVA 1.8 on your machine. Steps can be referred here : <https://www3.ntu.edu.sg/home/ehchua/programming/howto/JDK_Howto.html>
* Install and setup maven:

<https://www.tutorialspoint.com/maven/maven_environment_setup.htm>

* Setup Eclipse :
  + For Maven - <https://www.toolsqa.com/java/maven/how-to-install-maven-eclipse-ide/>
  + For TestNg - <https://www.ecanarys.com/Blogs/ArticleID/169/How-to-Install-TestNG-framework-Step-by-Step-installation-process>

### Project Setup

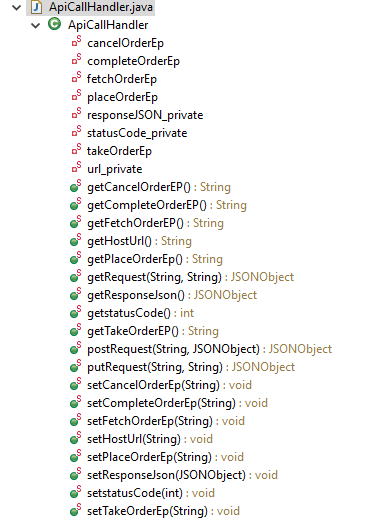
* Download the project folder from GitHub.
* In Eclipse import this project as existing maven project
* Right click on Project ->Mave->Update Project

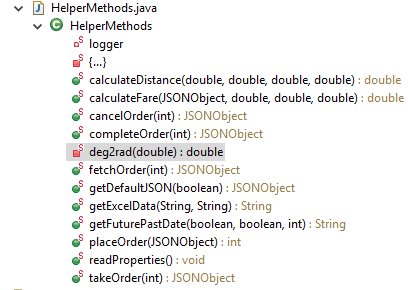
## Project Structure

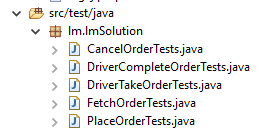
### Code Structure

Following is the code structure in the project :

* Src/main/java – This folder consists of following classes
  + APICallHandler.java – contains methods to handler the POST,PUT and GET request against the endpoint



* + Helper methods.java – contains methods to place Order, take Order etc which could be directly used in the test cases. 
  + ExcelReader.java – This is the utility class containing methods to read excel data.
* Src/test/java -- This folder contains Test classes for different API operations and a total of 37 test cases.



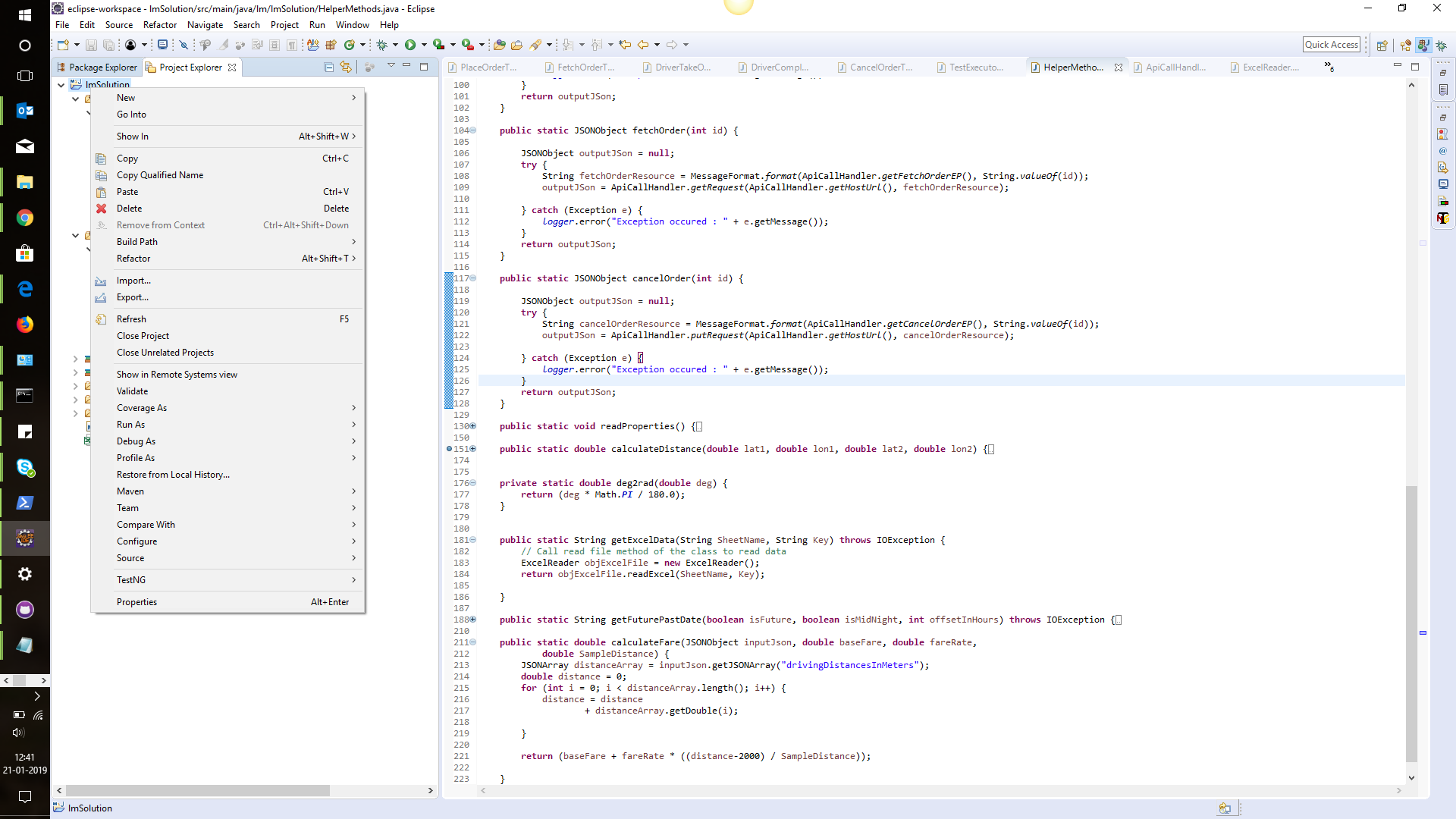
* Test Executor.java – This class contains the main method which executes the testing test cases when executed as runnable jar.

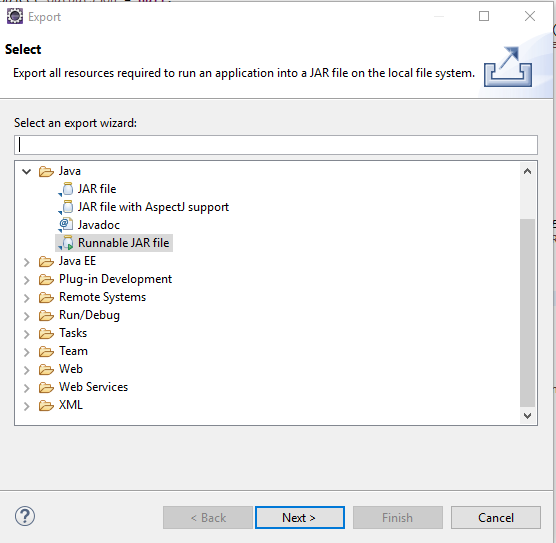
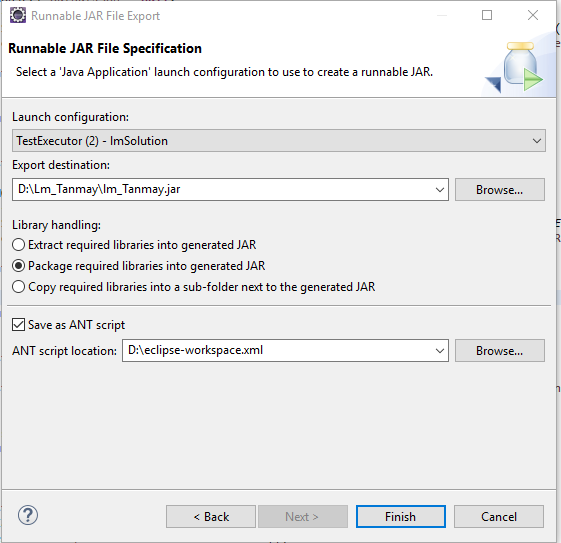
### Configuration, test data and logging

* **Config.Properties** file is used to configure some of the global configuration such as app url, endpoints, path of test data file.
* **Test Data** – Test data to be used by test cases have been saved in .xls file containing 3 sheets as of now:
  + TestCaseParams – This sheet contains the test case wise JSON payload data.
  + ErrorMessages – This sheet contains different error messages for various scenarios covered in test cases.
  + TestData – This sheet contains additional variable data which is used in the test cases.
* **Logging –** Log4j library is used for logging. Log4jlogger could be used to modify the level of logging as well as to change the path of output log file.

## Steps to create runnable jar

* Right click on the project and select Export.



* The wizard to export opens. In that window. Select Java -> Runnable jar file. Click Next
* In the Next screen, configure as follows :   
  Launch of configuration – Select the configuration containing the class having the main() method. This will be the starting point of the runnable jar.  
  Export destination – Select the directory to export to. Click Finish.
* Go to the destination where export is done. Executable jar would be present there.
* Copy and paste the following files from your project folder to jar location as they would be required for execution by the JAR file :
  + Config.properties
  + Log4j.properties
  + TestData.xls

## Test Case Execution

Test case execution could be done in the following ways:

* **Using TestNg from Eclipse IDE**: In Eclipse, right click on the **testing.xml** file in the project and select TestNG 🡪 Run As 🡪 TestNg Suite. All the test classes specified in the testing.xml file will be executed.
* **Using the Runnable Jar** :
  + Create runnable jar as described in previous section
  + Go to the JAR directory and from there open the command prompt and run the following command : java -jar *<JarfileName.jar>*

## Reports

* In case of both above specified executions there would be “**test-output**” folder created containing the testing reports for the previous execution
* Along with it there would be “**log4j-application.log**” containing the log4j logs for the execution. The name of this file could be changes using the log4j.properties file.

## Test scenarios covered

