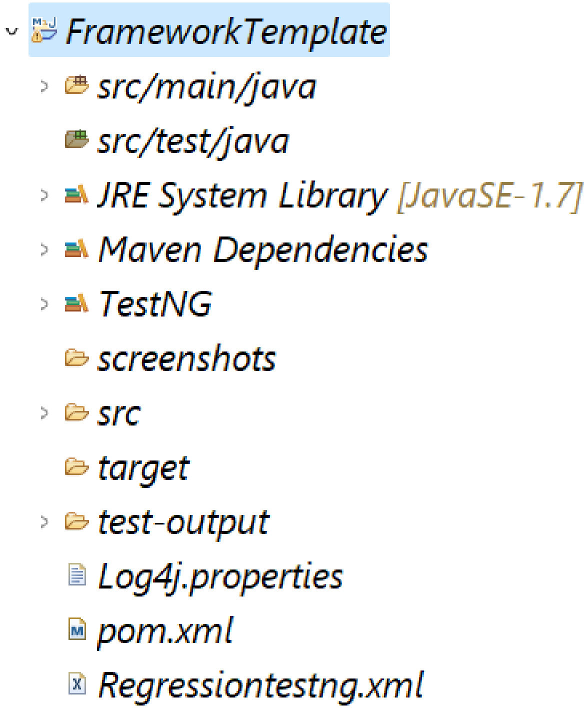
Maven Project -

- Maven is software project management tool and build automation tool.
- It is hosted by the Apache Software Foundation in 13 July 2004. It was begin with the Jakarta Project.
- Maven projects are configured using the POM (Project Object Model), which is stored in pom.xml file.
- Maven dynamically downloads the Java libraries, maven plugins from one or more repositories and store them in local cache.

Maven is used:

- To build the project
- To manage the project
- To define the project structure
- Test management
- Dependency build
- It is used to check the compilation issues between the framework components whenever the multiple test engineers integrate their files into same framework.

Maven Project structure:



```
FrameworkTemplate
    src/main/java
    src/test/java
    JRE System Library [JavaSE-1.7]
    Maven Dependencies
    TestNG
    screenshots
    src
    target
    test-output
    Log4j.properties
    pom.xml
    Regressiontestng.xml
```

So, there are two main folder that is main java (to write test script logic) and test java (to write test cases) and other are supporting plug-ins or files.

pom.xml file:

- A Project Object Model (pom) an XML file which have information about the project.
- It has different tabs like overview, dependencies, dependencies hierarchy, effective pom, pom.xml
    - In overview tab, there are artifact details and project details.
    - Details like group id, artifact id, version, project name, URL.
    - In dependencies tab, there are dependencies which added in our project. We can add it and remove it as per our requirement.
    - Dependencies hierarchy tab, where as its name says hierarchy of dependencies are there.
    - Effective pom tab, it is detailed xml file, all information about project you will find here.
- pom.xml tab, it has group id, artifact id, version details, Where we can add dependencies as
    - first we have to go on to the maven repository website and then search the dependency which we want then select the version of it just click on code it will automatically copy then paste it in pom.xml file.

Here's an example:

```
<project>
        <modelVersion>4.0.0</modelVersion>
        <groupId>com.mycompany.app</groupId>
        <artifactId>my-app</artifactId>
        <version>1</version>
</project>
```

- <project> –> it is the root
- <modelVersion> -> should be set to 4.0.0
- <groupId> -> The id of the project's group.
- <artifactId> -> The id of the artifact (project)
- <version> -> The version of the artifact under the specified group

src/main/java

- Here we are going to write code by using Page Object Model design pattern with Page Factory.
- POM: As per the Page Object Model, we have to maintain classes for every web page. Each web page has a separate class, and that class holds the functionality and members of that web page. Separate classes for every individual test.
- POM uses encapsulation feature of OOPs, (Explain Encapsulation in detail). where variables are private and methods are public and we use getter and setter methods but there is disadvantage of POM i.e. can't find the hidden elements so we will use page factory class to overcome disadvantage of POM
- In page factory we use @findby annotation above your web element and we have to use static method like init element to initialize the data member in page factory. which contains the locators, different methods to perform actions on web elements like sendKeys(), click(), etc.
- It also contains the Utility class, Base class, Property class

src/test/java

- As per the Maven project, all tests classes are kept in the 'src/test/java' folder.
- So it contains the execution for all the POM classes which are stored in src/main/java. Because in POM classes we can't write the main method.