Experiment No.7
To Setup and Run Selenium Tests in Jenkins Using Maven.
Date of Performance:
Date of Submission:

Aim: To Setup and Run Selenium Tests in Jenkins Using Maven

Objective: Objective is to setup enables seamless integration of automated testing into the CI/CD pipeline, facilitating faster feedback loops and promoting a culture of continuous improvement in software development.

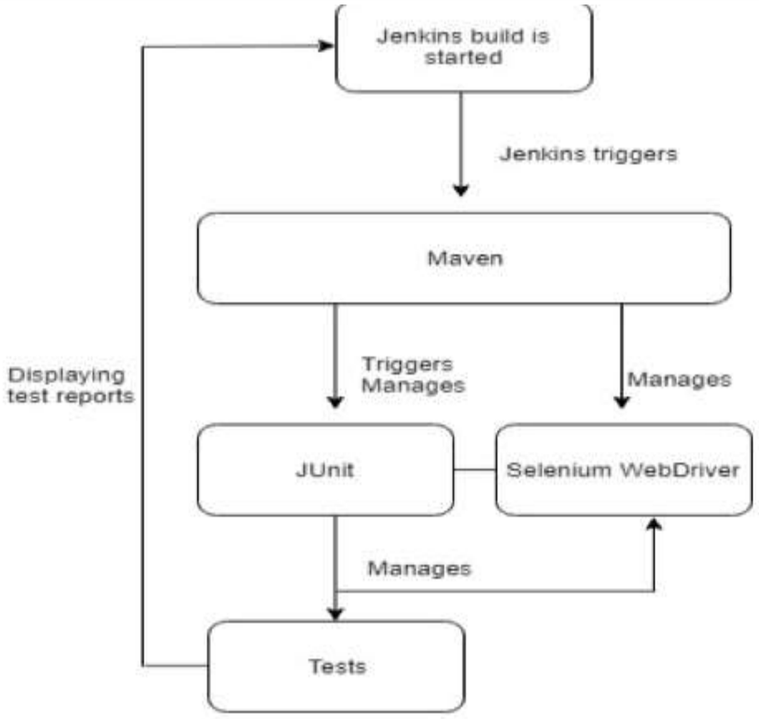
Theory:

Jenkins is the leading open-source continuous integration tool developed by Hudson lab. It is cross-platform and can be used on Windows, Linux, Mac OS and Solaris environments. Jenkins is written in Java. It has taken the place as one of the best open-source tools that allow continuous integration and build management.

Running Selenium tests in Jenkins allows you to run your tests every time your software changes and deploy the software to a new environment when the tests pass. Jenkins can schedule your tests to run at specific time. You can save the execution history and Test Reports. Jenkins supports Maven for building and Testing a project in continuous integration

Maven is a powerful project / build management tool, based on the concept of a POM (Project Object Model) that includes project information and configuration information for Maven such as construction directory, source directory, dependency, test source directory, Goals, plugins, etc.

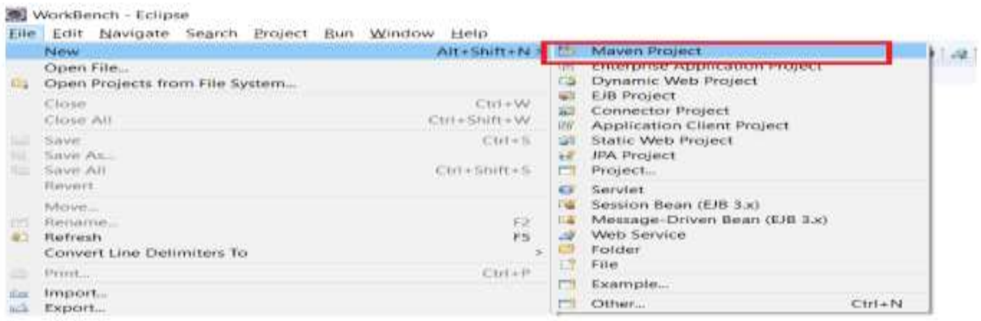
Integrating Maven with Selenium provides following benefits Apache Maven provides support for managing the full lifecycle of a test project. Maven is used to define project structure, dependencies, build, and test management. Using pom.xml(Maven) you can configure dependencies needed for building testing and running code. Maven automatically downloads the necessary files from the repository while building the project.



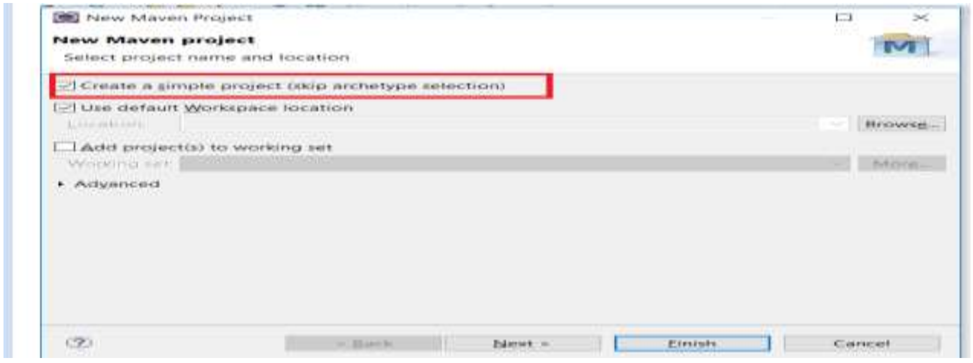
Steps:

---Create a Maven Selenium script---

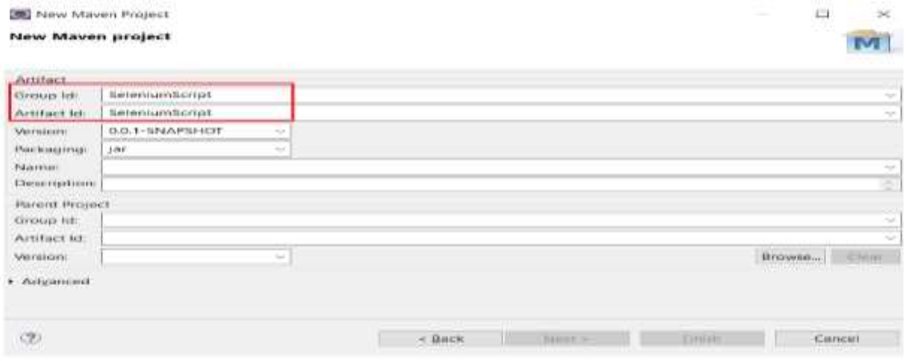
1. In Eclipse IDE, create a new project by selecting **File | New | Maven Project** from Eclipse menu.



2. On the **New Maven Project** dialog select the **Create a simple project** and click **Next**


New Maven Project
New Maven project
Select project name and location
Create a simple project (skip archetype selection)
Use default Workspace location
Location:
Browse...
Add project(s) to working set
Working set:
More...
Advanced
< Back
Next >
Finish
Cancel

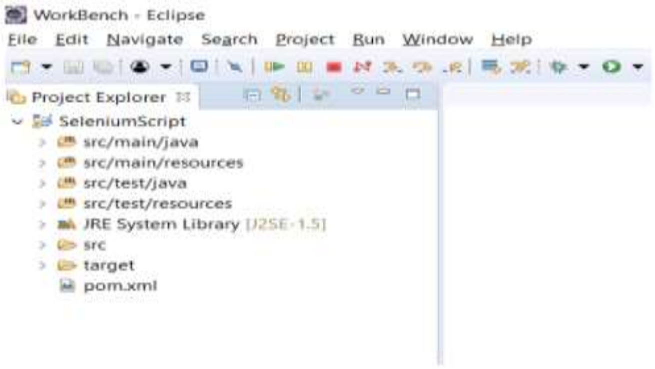
3. Enter **SeleniumScript** in **Group Id**: and **Artifact Id**: and click finish



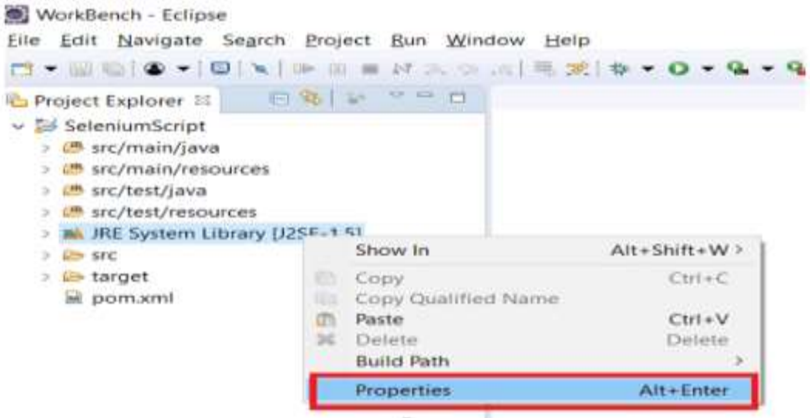
Group Id: SeleniumScript
Artifact Id: SeleniumScript
Version: 0.0.1-SNAPSHOT
Packaging: jar
Name:
Description:

Parent Project
Group Id:
Artifact Id:
Version:

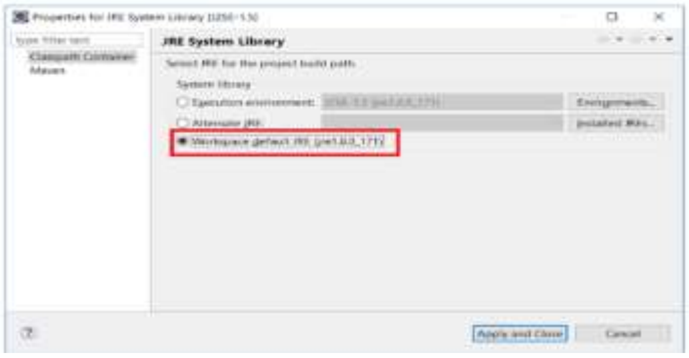
4. Eclipse will create webdriverTest.



5. Right click on JRE System Library and select the Properties option from the menu.



6. On the Properties for JRE System Library dialog box , make sure Workspace default JRE is selected and click ok.

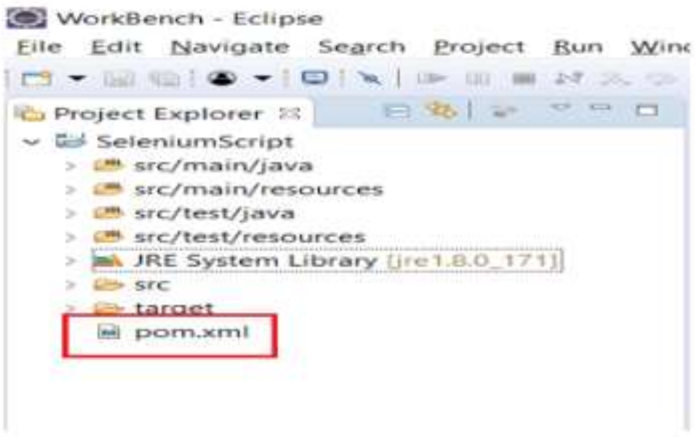


Select JRE for the project build path.

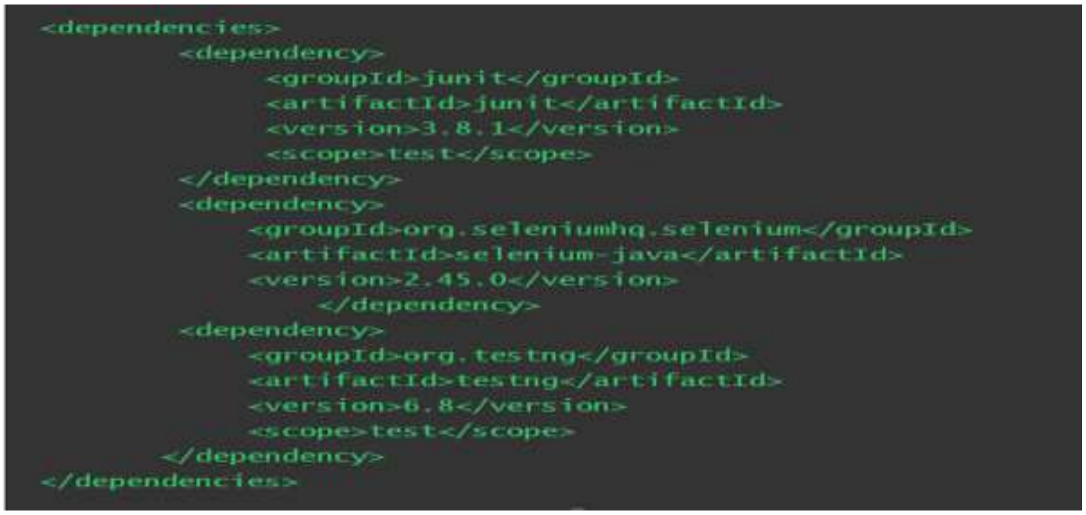
System library

- ( ) Execution environment: [JavaSE-1.8 (jre1.8.0_171)] Environments...
- ( ) Alternate JRE: Installed JREs...
- (●) Workspace default JRE (jre1.8.0_171)

7. Select pom.xml from project explorer.

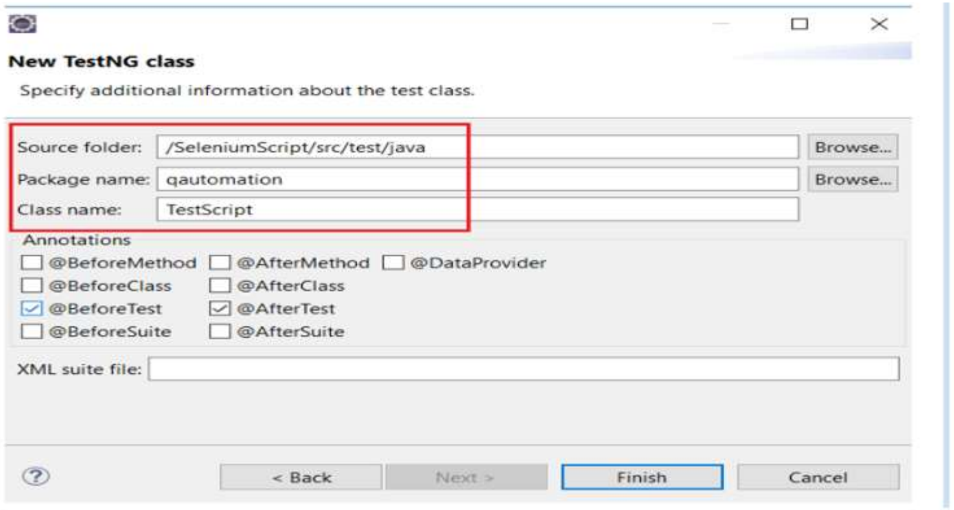


8. Add selenium, Maven, TestNG, Junit dependencies to pom.xml in the code.



```
<dependencies>
        <dependency>
             <groupId>junit</groupId>
             <artifactId>junit</artifactId>
             <version>3.8.1</version>
             <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>org.seleniumhq.selenium</groupId>
            <artifactId>selenium-java</artifactId>
            <version>2.45.0</version>
                </dependency>
        <dependency>
            <groupId>org.testng</groupId>
            <artifactId>testng</artifactId>
            <version>6.8</version>
            <scope>test</scope>
        </dependency>
</dependencies>
```

9. Create a new file TestNG class File|New|Others|TestNG|TestNG Class. Enter Package name as "Qautomation" and "TestScript" in the Name:textbox and click on the Finish button.


New TestNG class

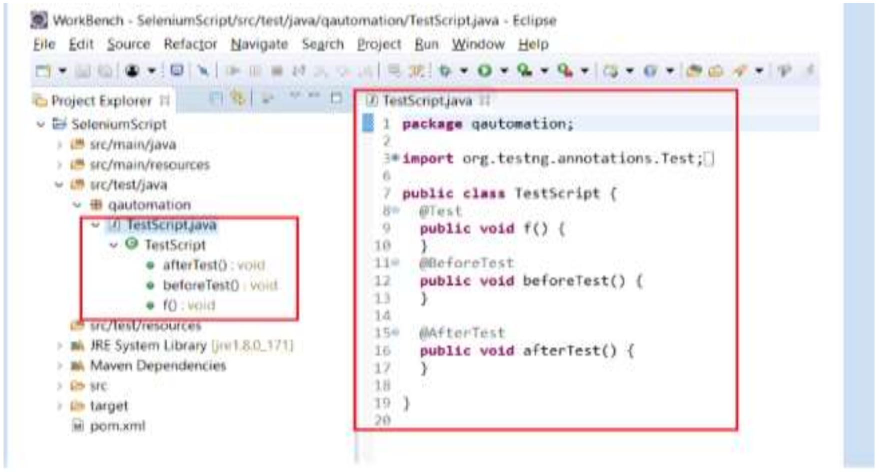
Specify additional information about the test class.

Source folder: /SeleniumScript/src/test/java
Package name: qautomation
Class name: TestScript

Annotations
- [ ] @BeforeMethod
- [ ] @AfterMethod
- [ ] @DataProvider
- [ ] @BeforeClass
- [ ] @AfterClass
- [✓] @BeforeTest
- [✓] @AfterTest
- [ ] @BeforeSuite
- [ ] @AfterSuite

XML suite file:

10. Eclipse will create the TestScript class


WorkBench - SeleniumScript/src/test/java/qautomation/TestScript.java - Eclipse

```
package qautomation;

import org.testng.annotations.Test;

public class TestScript {
  @Test
  public void f() {
  }
  @BeforeTest
  public void beforeTest() {
  }

  @AfterTest
  public void afterTest() {
  }

}
```

11. Add following code to the TestScript class and respective browser drivers for chrome,firefox and IE.

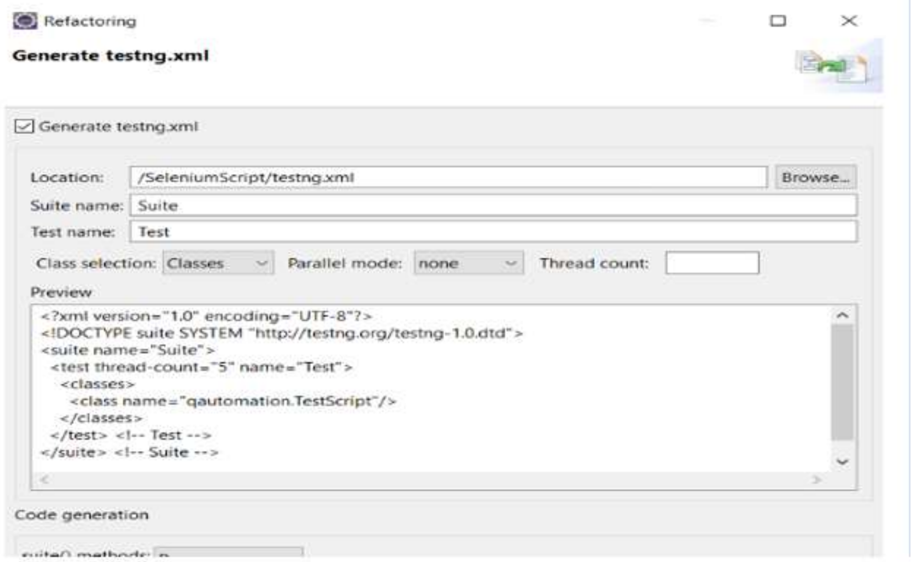
```
package qautomation;
import org.testng.annotations.Test;
import org.testng.annotations.BeforeTest;
import java.util.HashMap;
import java.util.Map;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.firefox.FirefoxOptions;
import org.openqa.selenium.firefox.FirefoxProfile;
import org.openqa.selenium.ie.InternetExplorerDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.testng.Assert;
import org.testng.annotations.AfterTest;
public class TestScript {
        public static WebDriver driver=null;
        public String browser = System.getProperty("browser");
        public String url = System.getProperty("URL");

 @BeforeTest
  public void beforeTest() {

        if(browser.equalsIgnoreCase("Chrome"))
        {
        System.setProperty("webdriver.chrome.driver",
        System.getProperty("user.dir")+"\\chromedriver.exe");
        Map<String, Object> prefs = new HashMap<String, Object>();
        ChromeOptions options = new ChromeOptions();
        options.setExperimentalOption("prefs", prefs);
        options.addArguments("--disable-arguments");
        options.addArguments("--test-type");
        options.addArguments("test");
        options.addArguments("disable-infobars");
        driver = new ChromeDriver(options);
        }
        else if(browser.equalsIgnoreCase("FireFox"))
        {
        System.setProperty(FirefoxDriver.SystemProperty.DRIVER_USE_MARIONETTE
        ,"true");
        System.setProperty(FirefoxDriver.SystemProperty.BROWSER_LOGFILE,Syste
        m.getProperty("user.dir")+"\\FireFoxLogs.txt");
        System.setProperty("webdriver.gecko.driver",
        System.getProperty("user.dir")+"\\geckodriver_v23.exe");
        FirefoxProfile profile = new FirefoxProfile();
        profile.setAcceptUntrustedCertificates(false);
        FirefoxOptions options = new FirefoxOptions().setProfile(profile);
        driver = new FirefoxDriver(options);
        driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
        driver.manage().window().maximize();
        }
        else if (browser.equalsIgnoreCase("IE"))
        {
        System.setProperty("webdriver.ie.driver",
        System.getProperty("user.dir")+"\\IEDriverServer351.exe");
        DesiredCapabilities caps = DesiredCapabilities.internetExplorer();
```

```
            caps.setCapability(InternetExplorerDriver.INTRODUCE_FLAKINESS_BY_IGNO
            RING_SECURITY_DOMAINS,true);
            caps.setCapability(InternetExplorerDriver.IGNORE_ZOOM_SETTING,true);
            caps.setCapability(InternetExplorerDriver.UNEXPECTED_ALERT_BEHAVIOR,"
            accept");
            caps.setCapability(InternetExplorerDriver.REQUIRE_WINDOW_FOCUS,true);
            caps.setCapability(InternetExplorerDriver.INITIAL_BROWSER_URL,"http://www.
            google.com/");
            driver = new InternetExplorerDriver(caps);
            driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
            driver.manage().window().maximize();
            }
            driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
            driver.manage().window().maximize();
      }
     @Test
     public void TestApplication() {
     driver.get(url);
     String title = driver.getTitle();
     System.out.println("Title="+title);
     Assert.assertTrue(title.contains("QAutomation"));
      }
     @AfterTest
      public void afterTest() {
            driver.quit();
      }
}
```

12. Right click on the WebdriverTest and select TestNG| Convert to TestNG. Eclipse will create testing.xml which says that you need to run only one test with the name TestApplication.


Refactoring

Generate testng.xml

☑ Generate testng.xml

Location: /SeleniumScript/testng.xml Browse...

Suite name: Suite

Test name: Test

Class selection: Classes Parallel mode: none Thread count:

Preview

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Suite">
  <test thread-count="5" name="Test">
    <classes>
      <class name="qautomation.TestScript"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

Code generation

13. Adding dependencies and plugins

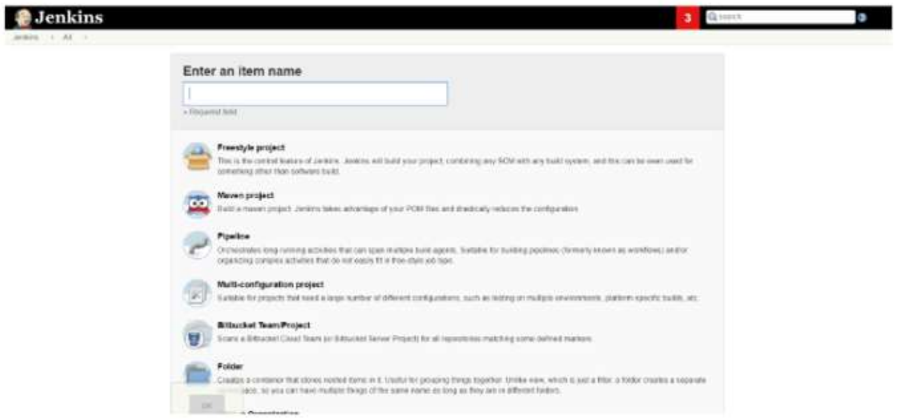
    Additionally we need to add

    1. Maven-compiler-plugin
    2. Maven-surefire-plugin
    3. Testng.xml

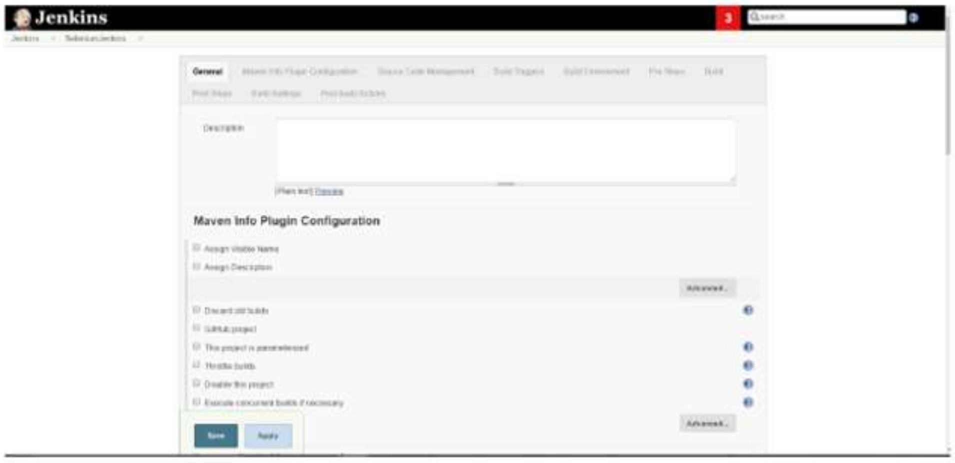
---------Integrating your test to Jenkins-------------

1. Launch and login into jenkins URL – http://localhost:8080/

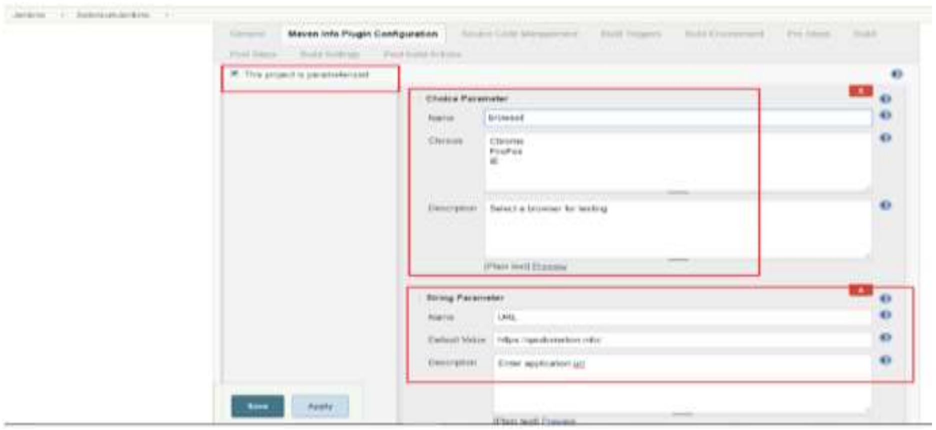
2. Click on new item and enter an appropriate name for the new job , select Maven Project and click on save.



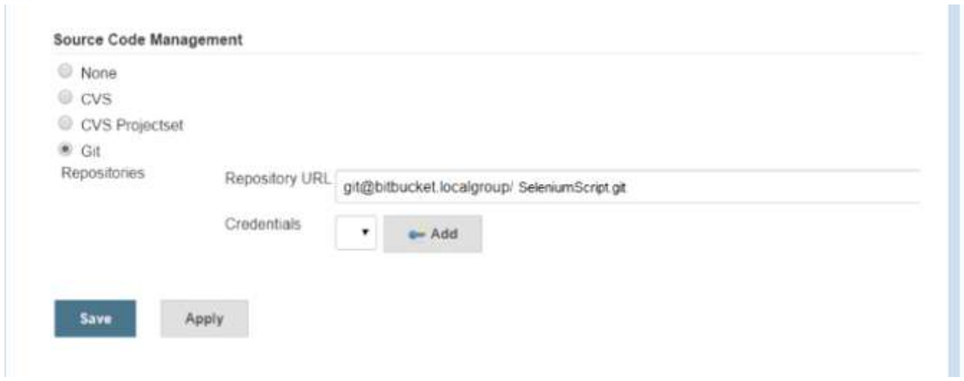
3. A new empty job has been created at this point.



4. Jenkins Parameterized Build in Jenkins just check the checkbox **This project is paramerized** and add the parameter by **Add Parameter** as per your project requirement.



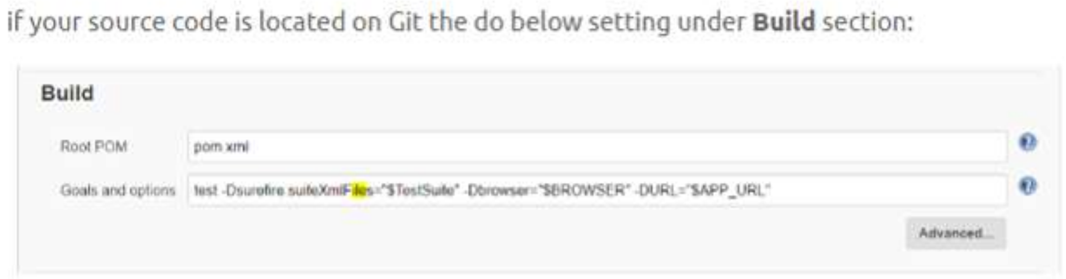
5. If code is located on Git Under **Source Management** , select the appropriate repository for the location of project and pass the URL and credentials.


Source Code Management

- None
- CVS
- CVS Projectset
- [✓ Git]

Repositories
Repository URL: git@bitbucket.localgroup/ SeleniumScript.git
Credentials

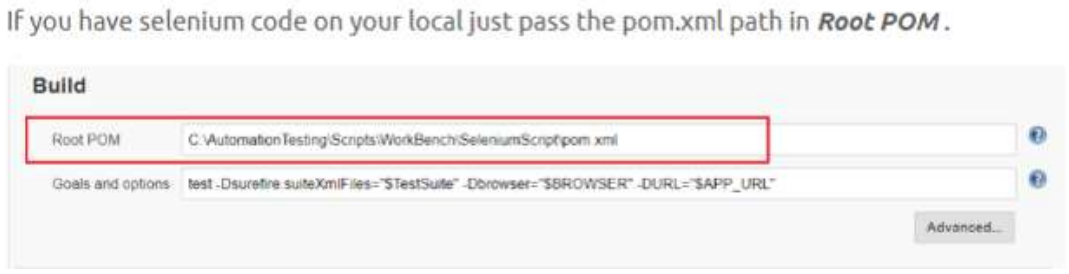
6. In the “pre-steps” build section another set of parameters can be passed to the Jenkins build. Specify the Maven targets that need to be executed in order to run test.


if your source code is located on Git the do below setting under **Build** section:

Build

Root POM: pom.xml

Goals and options: test -Dsurefire.suiteXmlFiles="$TestSuite" -Dbrowser="$BROWSER" -DURL="$APP_URL"

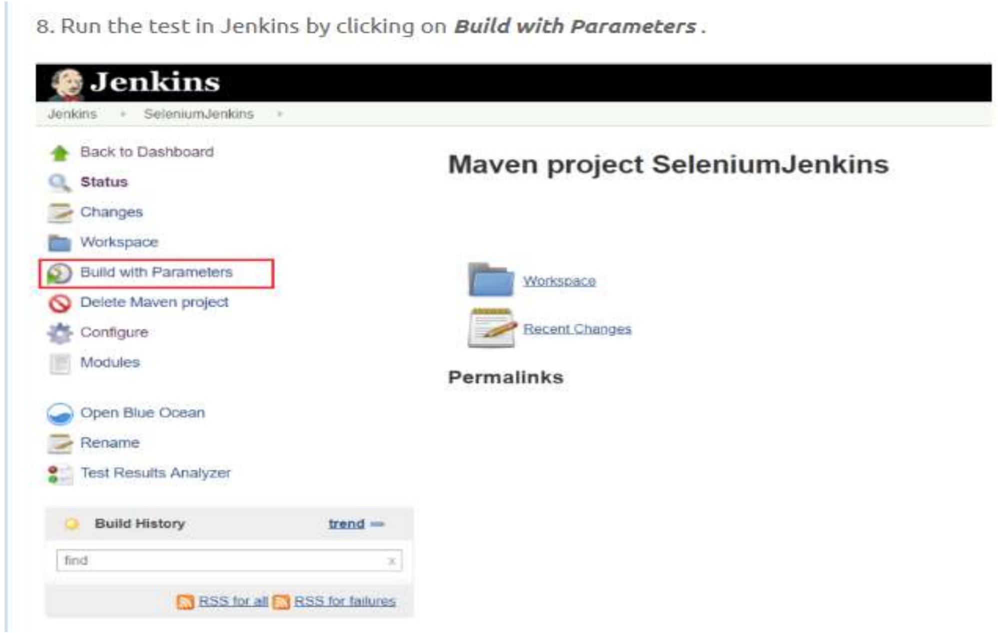

If you have selenium code on your local just pass the pom.xml path in ***Root POM*** .

Build

Root POM: C:\AutomationTesting\Scripts\WorkBench\SeleniumScript\pom.xml

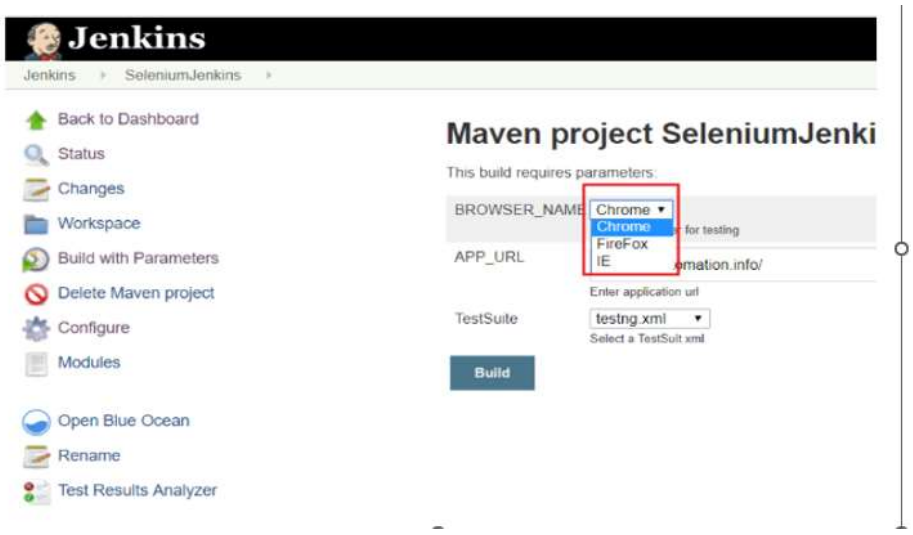
Goals and options: test -Dsurefire.suiteXmlFiles="$TestSuite" -Dbrowser="$BROWSER" -DURL="$APP_URL"

7. Run the test in Jenkins by clicking on Building with Parameters.

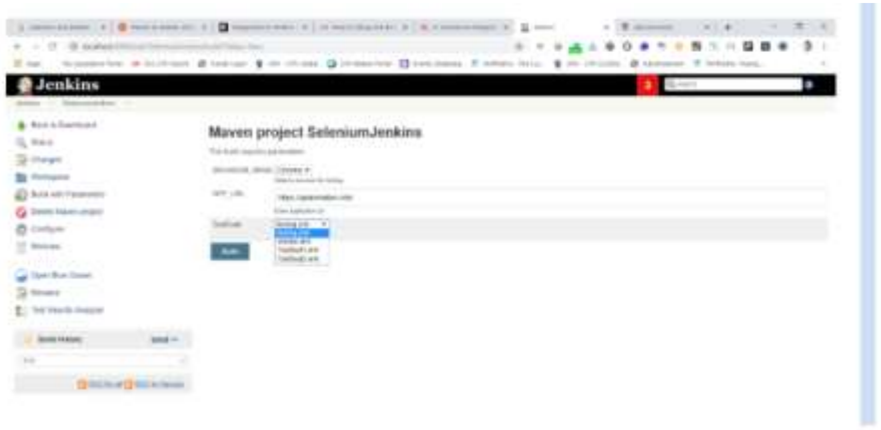


8. Run the test in Jenkins by clicking on ***Build with Parameters*** .

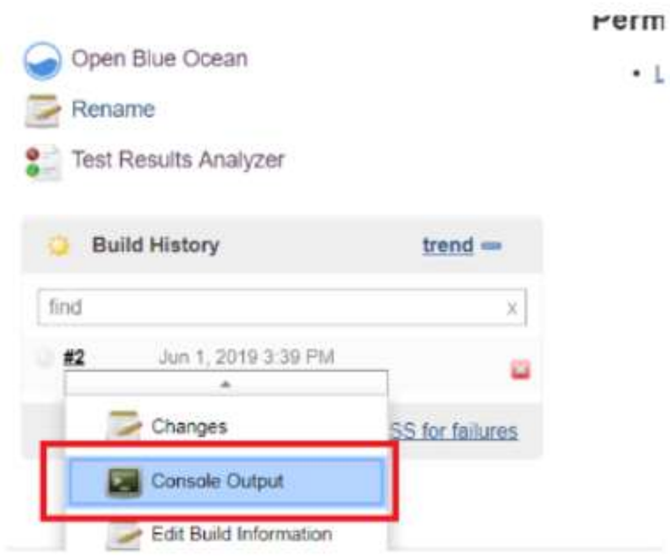
8. Select the browser you want to run from dropdown.



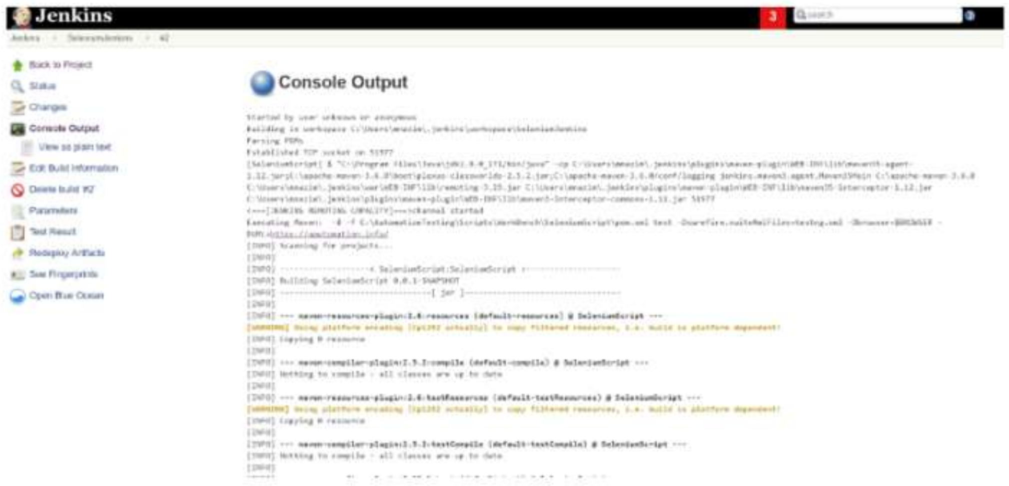
9. Select the TestSuit file.



10. Click the build button and go to console output .



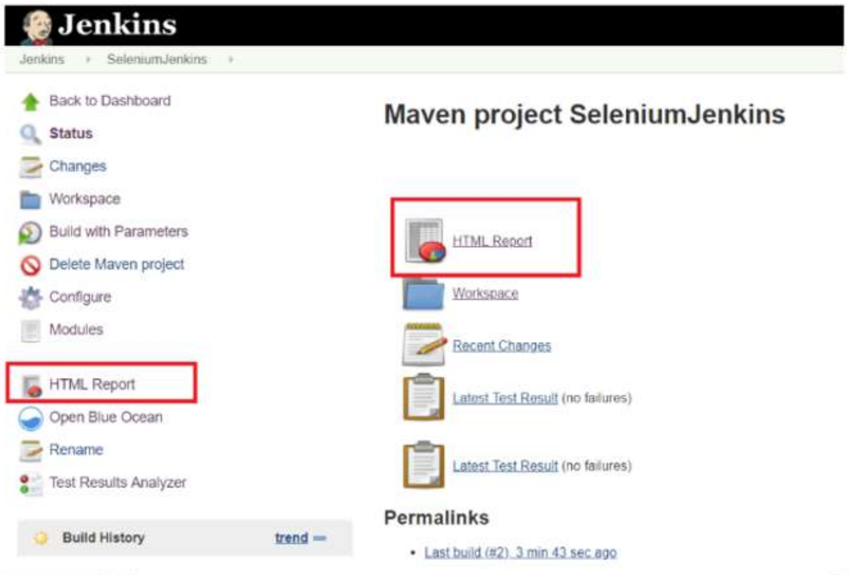
11. See the logs from **Console Output** window.



```
Started by user unknown or anonymous
Building in workspace C:\Users\[illegible]\.jenkins\workspace\SeleniumJenkins
Parsing POMs
Established TCP socket on 51977
[SeleniumScript] $ "C:\Program Files\Java\jdk1.8.0_171/bin/java" -cp C:\Users\[illegible]\.jenkins\plugins\maven-plugin\WEB-INF\lib\maven35-agent-
1.12.jar;C:\apache-maven-3.6.0\boot\plexus-classworlds-2.5.2.jar;C:\apache-maven-3.6.0/conf/logging jenkins.maven3.agent.Maven35Main C:\apache-maven-3.6.0
C:\Users\[illegible]\.jenkins\war\WEB-INF\lib\remoting-3.35.jar C:\Users\[illegible]\.jenkins\plugins\maven-plugin\WEB-INF\lib\maven35-interceptor-1.12.jar
C:\Users\[illegible]\.jenkins\plugins\maven-plugin\WEB-INF\lib\maven3-interceptor-commons-1.12.jar 51977
<===[JENKINS REMOTING CAPACITY]===>channel started
Executing Maven:  -B -f C:\AutomationTesting\Scripts\WorkBench\SeleniumScript\pom.xml test -Dsurefire.suiteXmlFiles=testng.xml -Dbrowser=[illegible] -
POM: <https://automation.info/>
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------< SeleniumScript:SeleniumScript >-------------------
[INFO] Building SeleniumScript 0.0.1-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ SeleniumScript ---
[WARNING] Using platform encoding (Cp1252 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.3.2:compile (default-compile) @ SeleniumScript ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ SeleniumScript ---
[WARNING] Using platform encoding (Cp1252 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.3.2:testCompile (default-testCompile) @ SeleniumScript ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
```

Note : Blue color of ball of console output is that build is successful

12. View the html report just click on the link.


Jenkins
Jenkins › SeleniumJenkins ›

- Back to Dashboard
- Status
- Changes
- Workspace
- Build with Parameters
- Delete Maven project
- Configure
- Modules
- HTML Report
- Open Blue Ocean
- Rename
- Test Results Analyzer

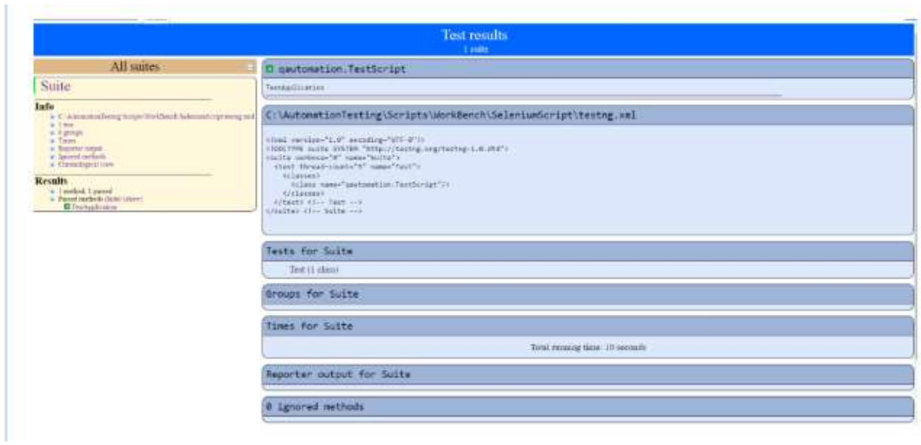
Build History trend

Maven project SeleniumJenkins

- HTML Report
- Workspace
- Recent Changes
- Latest Test Result (no failures)
- Latest Test Result (no failures)

Permalinks

- Last build (#2), 3 min 43 sec ago


Test results
1 suite

All suites
Suite
Info
Results
- 1 method, 1 passed

qautomation.TestScript
TestApplication

C:\AutomationTesting\Scripts\WorkBench\SeleniumScript\testng.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite verbose="0" name="Suite">
  <test thread-count="5" name="Test">
    <classes>
      <class name="qautomation.TestScript"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

Tests for Suite
Test (1 class)

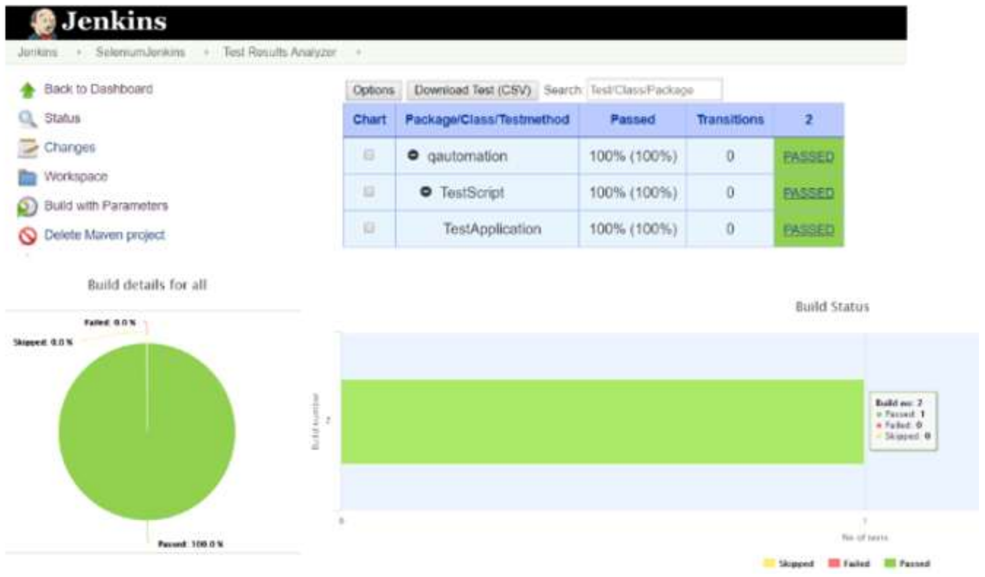
Groups for Suite

Times for Suite
Total running time: 10 seconds

Reporter output for Suite

0 ignored methods

13. Click Test Analyzer to analyse the result.


Jenkins
Jenkins › SeleniumJenkins › Test Results Analyzer ›

- Back to Dashboard
- Status
- Changes
- Workspace
- Build with Parameters
- Delete Maven project

Options | Download Test (CSV) | Search: Test/Class/Package

Chart	Package/Class/Testmethod	Passed	Transitions	2
☐	qautomation	100% (100%)	0	PASSED
☐	TestScript	100% (100%)	0	PASSED
☐	TestApplication	100% (100%)	0	PASSED

Conclusion:

Q1. Which browsers are supported by selenium webdriver?

Q2. What are some features of selenium 4?