

Contents

Selenium	4
Selenium Suite Components	4
Selenium IDE	4
Selenium RC	4
Selenium WebDriver	4
Selenium Grid	4
Why Selenium	4
Limitations of selenium.....	4
How Selenium WebDriver works	4
Selenium Architecture	5
Selenium First Program.....	5
Required software for Selenium Program.....	5
Steps to create Selenium Program.....	5
First Program	6
Browser Developer Tools	6
Browser Console	6
Object Identification	7
Selenium By Class.....	7
WebElement Class	9
Actions Class	10
JavascriptExecutor	11
Scroll Up and Scroll Down using JavascriptExecutor	11
Move to Element using JavascriptExecutor.....	11
Working with Edit / Text box	11
Working with check box.....	12
Working with option button	12
Working with Dropdown list box	12
Working with Web Table	13
Working with Frames.....	13
Capture Screenshot	15
Navigation Commands.....	15

Alerts.....	15
Selenium Wait.....	15
Browser Driver Options.....	16
Page Object Model.....	17
Page Factory Design Pattern	17
Maven.....	17
Maven Lifecycle Phases	17
Plug-ins vs Dependencies.....	17
Important points in Maven	18
WebDriverManager	20
WebDriverManager Maven Dependency.....	20
Selenium Server Maven Dependency	20
TestNG	20
TestNG Maven Dependency	21
TestNG Annotations.....	21
TestNG Reports.....	21
Attach Screenshot.....	21
TestNG Assertions.....	21
TestNG Listeners	21
Listener Example:.....	22
TestNG Suite XML	24
TestNG Parallel Execution	24
Maven TestNG Integration – SureFirePlugin.....	25
Maven Force Update Project	26
Run Maven Test	28
GIT	28
GIT Bash	28
Code / Steps push code into Github from Git Bash.....	29
Jira	30
Issue types in JIRA:.....	30
BDD.....	30
Pre-requisites.....	30
Cucumber Dependencies POM	32

Selenium Cucumber Project Explorer	32
Feature File	33
Step Definition File.....	33
Test Runner File	34
SQL.....	34
DDL – Data Definition Language	34
DML – Data Manipulation Language.....	34
DCL – Data Control Language.....	34
API Automation.....	34
HttpClient Maven Dependency.....	34
Implementation	35
Output	36
SLF4J (Log4J)	36
Dependencies	36
Implementation	37
Output	37

Selenium

Selenium is one of the most widely used open source Web UI automation testing tools. It supports automation testing of websites across different browsers such as Edge, Chrome, Safari and Firefox etc. It supports various platforms and programming languages. User can automate functional tests and easily integrates them with Maven, Jenkins and other build automation and continuous integration tools.

Selenium Suite Components

Selenium IDE

It is an add-on to Firefox browser that provides record and playback capability. It has limited features and test scripts generated are not robust and portable.

Selenium RC

Selenium Remote Control (RC) is officially deprecated by Selenium. It uses a dedicated server for running automation scripts.

Selenium WebDriver

It provides different drivers for different browsers and support different programming languages.

Selenium Grid

It helps in running selenium test scripts in parallel mode across multiple remote machines.

Why Selenium

1. Open Source
2. Scripting can be done in many programming languages like C#, Java, Python, Perl and Ruby etc.
3. Supports all popular browsers like Edge, Firefox, Chrome, IE, Opera and Safari etc.
4. Selenium grid helps in parallel and distributed test execution.

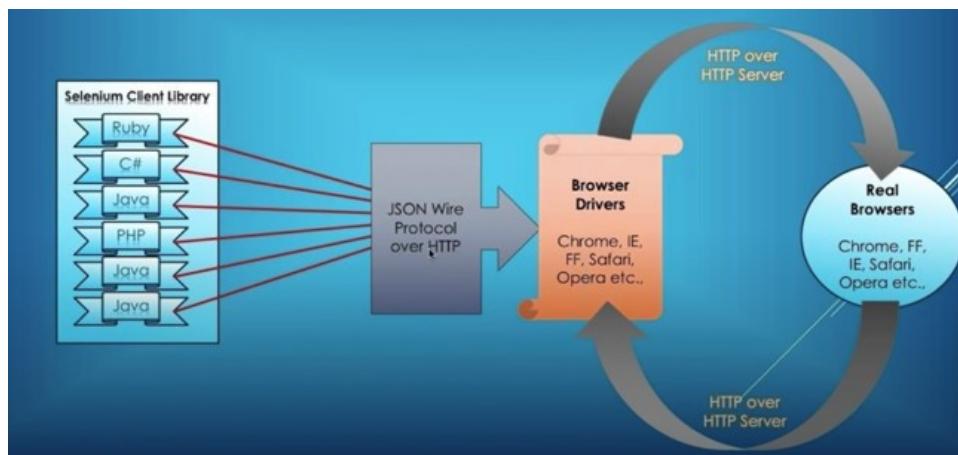
Limitations of selenium

1. Selenium does not support desktop application test automation. User need to use Robot Class or AutoIT for performing any windows based actions as part of Web Automation testing.
2. REST or SOAP web services cannot be automated
3. For reporting / logging, reading-writing external files rely on external libraries (TestNG, Extent Report, Allure Reports and POI Jars for excel file reading etc.)

How Selenium WebDriver works

Selenium WebDriver (Selenium 2.0) automates the browsers actions by calling their native events (Click, MouseOver, KeyPress etc.) directly unlike Selenium RC which injects Javascript in browsers for test automation. WebDriver is much faster than Selenium RC and also handles scenarios like alerts, pop-ups, Keyboard and Mouse actions easily. Some of the widely used drivers in Selenium are ChromeDriver, FirefoxDriver, InternetExplorerDriver, SafariDriver and HtmlUnitDriver (headless driver) etc.

Selenium Architecture



About JSON Wire Protocol

https://www.selenium.dev/documentation/legacy/json_wire_protocol/

Selenium First Program

Required software for Selenium Program

1. Install Java
2. Download Eclipse (ensure Maven plugin is available)

Additional details:

Selenium JAR from <https://www.selenium.dev/downloads/>

Browser Driver from <https://chromedriver.chromium.org/>

Steps to create Selenium Program

1. Open Eclipse and create a simple Maven Project
2. Update POM.xml with Selenium-Java and WebDriverManager dependency

```
OnlinePortal/pom.xml
1<project xmlns="http://maven.apache.org/POM/4.0.0"
2   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
4     <modelVersion>4.0.0</modelVersion>
5     <groupId>walmart</groupId>
6     <artifactId>OnlinePortal</artifactId>
7     <version>0.0.1-SNAPSHOT</version>
8     <dependencies>
9       <!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
10      <dependency>
11        <groupId>org.seleniumhq.selenium</groupId>
12        <artifactId>selenium-java</artifactId>
13        <version>4.3.0</version>
14      </dependency>
15      <!-- https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager -->
16      <dependency>
17        <groupId>io.github.bonigarcia</groupId>
18        <artifactId>webdrivermanager</artifactId>
19        <version>5.2.1</version>
20      </dependency>
21    </dependencies>
22  </project>
```

First Program

```
J SeleniumDemo.java X
1 package day23;
2
3 import org.openqa.selenium.WebDriver;
4 import org.openqa.selenium.chrome.ChromeDriver;
5 import org.openqa.selenium.edge.EdgeDriver;
6
7 import io.github.bonigarcia.wdm.WebDriverManager;
8
9 public class SeleniumDemo {
10     public static void main(String[] args) {
11         WebDriverManager.chromedriver().setup();
12         WebDriver driver = new ChromeDriver();
13         driver.get("https://www.google.co.in");
14         driver.close();
15         driver.quit();
16
17         WebDriverManager.edgedriver().setup();
18         WebDriver driver2 = new EdgeDriver();
19         driver2.get("https://www.intel.com");
20         driver2.close();
21         driver2.quit();
22     }
23 }
```

Notes:

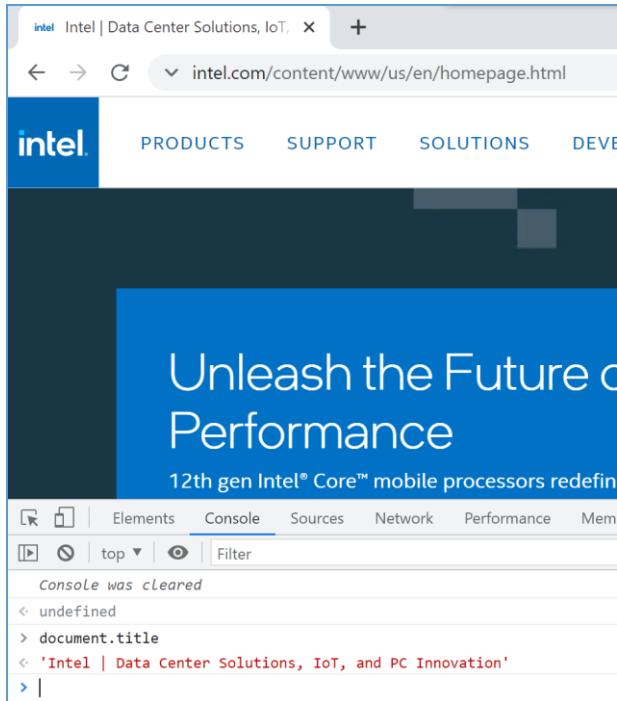
1. WebDriver is an interface. It has empty methods but not implemented. These methods can be implemented anyone until method type and signatures are not violated.
2. ChromeDriver is a class that has been written specially for Chrome Browser. It has many methods and implemented and can be instantiated. ChromeDriver can perform all actions on the Chrome browser as defined in the interface WebDriver.

Browser Developer Tools

Use F12 function key to get a Browser Developer tools.

Browser Console

Click on Console Tab to give any browser (Document) related commands like below:



Object Identification

Selenium By Class

Selenium By Class used to locate elements within a document using className, cssSelector, equals, hashCode, id, linkText, name, partialLinkTest, tagName and [xpath](#).

Note: Ensure your xpath is able to identify your application WebElement uniquely.

XPath

XPath(XML Path Language) is based on a tree representation of the XML document and used for selecting nodes the document. It also provides ability to navigate around the tree.

1. Absolute XPath – A single "/" at the beginning instructs XPath engine start looking for an element from root node.

/html/body/div[2]/div/div/footer/section[3]/div/ul/li[3]/a

2. Relative XPath – A double "//" instructs XPath engine look for an element anywhere in the XML document for given criteria.

Syntax: //TagName[@AttributeName='Attribute Value']

Examples:

//*[@id='lst-ib']

//button[@label='Continue']

3. Partial XPath – If locators properties changing frequently then try with “Contains” keyword.

//div[contains(@class, 'footer-nav')]

//h1[text()='Mobile Deposit Confirmation']

Partial XPath Keyword

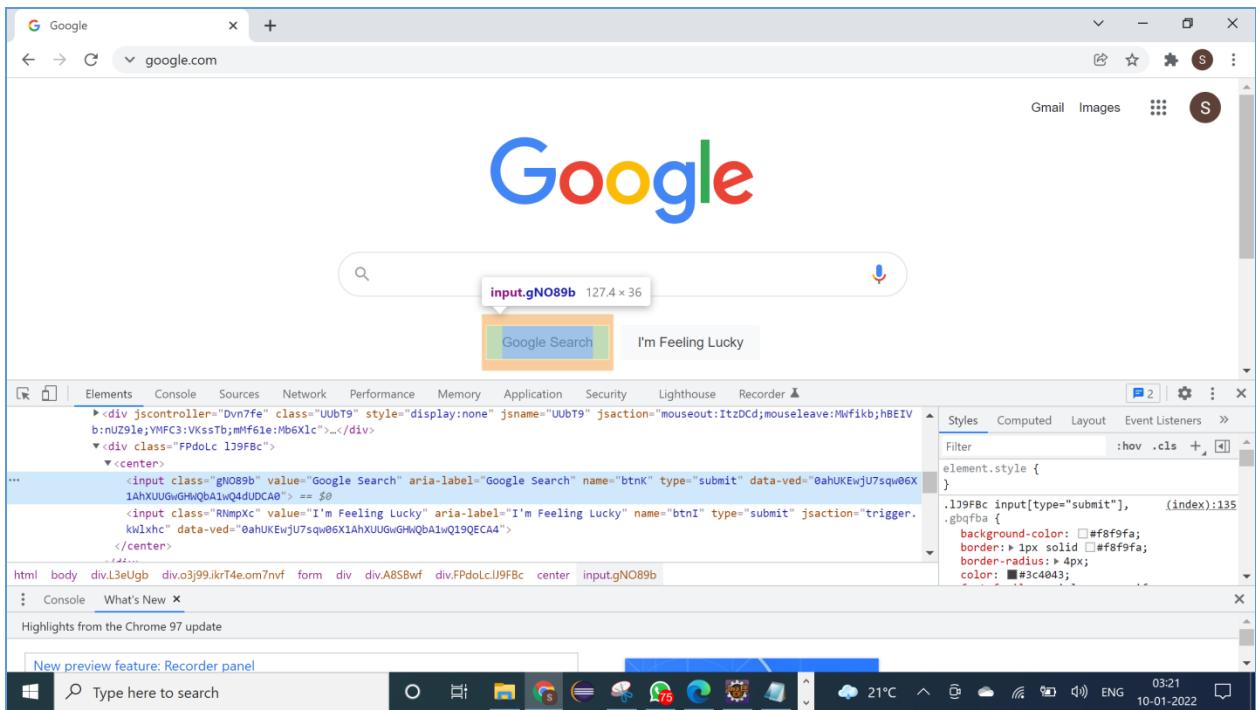
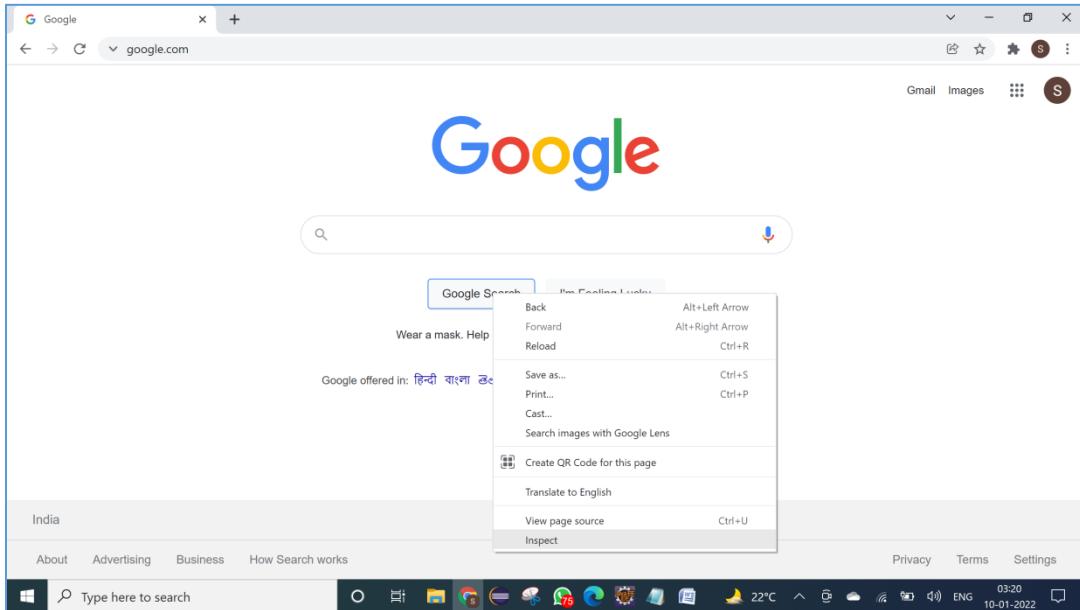
1. Contains
2. Text

3. Start-with

DOM – Document Object Model

To write xpath of any WebElement in a Chrome browser follow below steps:

1. Open browser and launch your application URL
2. Mouse Hover on WebElement you want to get xpath
3. Mouse Right click on the WebElement and click on Inspect from shortcut menu
4. You will get DOM of the webpage



XPath Axes

The screenshot shows a browser's developer tools with the DOM tree expanded. A yellow box highlights the current node, which is an input element with id="u_0_3". An arrow points from the text "Current node" to this yellow box. Another arrow points from the text "Grandparent of current node" to the parent node, which is a td element. A third arrow points from the text "XPath using ancestor axis" to the XPath expression //input[@id='u_0_3']/ancestor::label in the address bar.

DOM Tree:

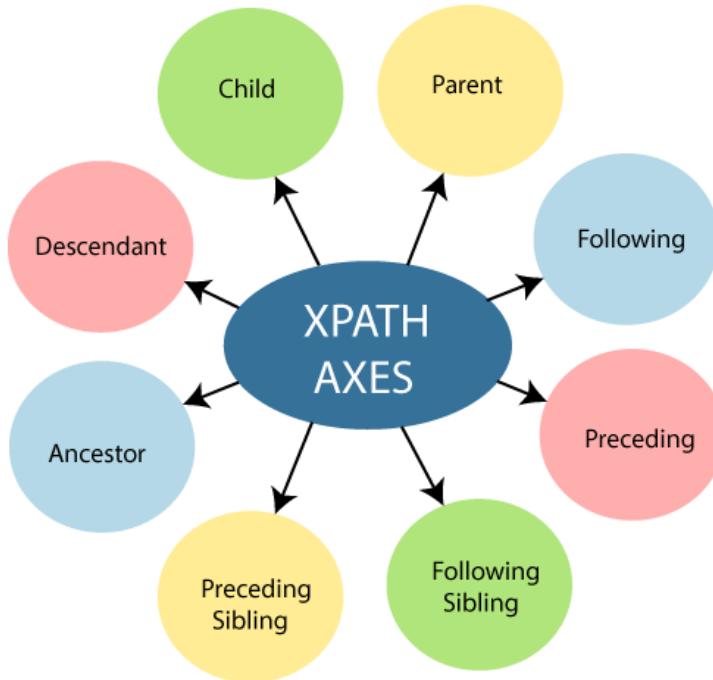
```
<tr>
  <td>...</td> Parent node of current node
  <td>...</td>
  <td>
    <label class="uiButton uiButtonConfirm" id="loginbutton" for="u_0_3">
      <input value="Log In" aria-label="Log In" data-testid="royal_login_button" type="submit" id="u_0_3" /> == $0
    </label>
  </td>
</tr>
```

Address Bar:

```
//input[@id='u_0_3']/ancestor::label
```

Toolbar:

```
1 of 1 ▲ ▼ Cancel
```



WebElement Class

Anything that is present on the web page is a WebElement such as text box, button, etc. WebElement represents an HTML element. Selenium WebDriver encapsulates a simple form element as an object of the WebElement. It basically represents a DOM element and all the HTML documents are made up by these HTML elements. Each HTML element consists of a start tag and an end tag. The content lies between the tags.

```
WebElement searchBox = driver.findElement(By.name("q"));
```

In above example, `searchBox` is a `WebElement` and user can perform any below actions:

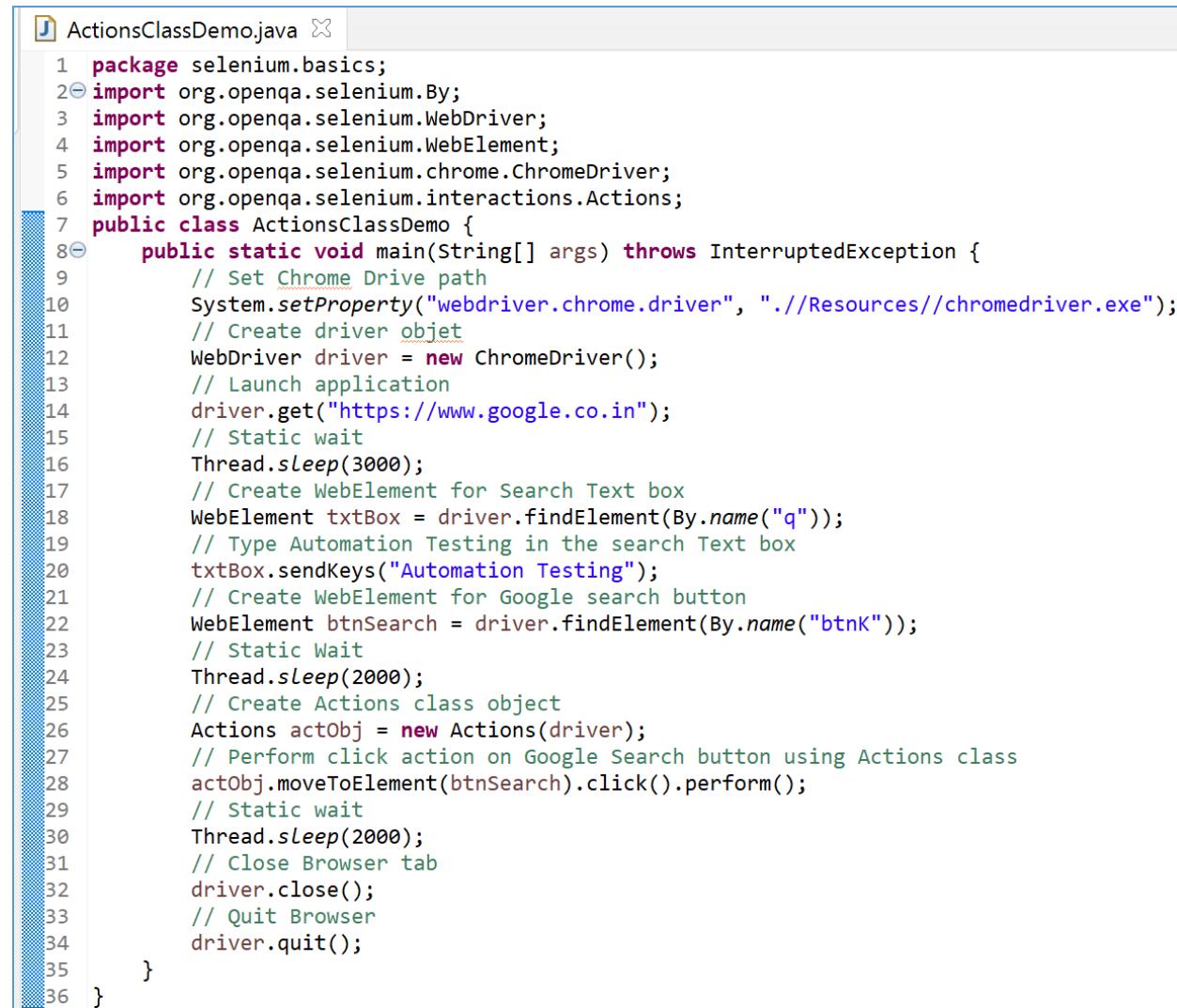
1. `.sendKeys`
2. `.isDisplayed`
3. `.isSelected`
4. `.submit`
5. `.isEnabled`
6. `.clear`
7. `.click`

[Refer this link for more WebElement commands / actions.](#)

[Refer this link to understand various WebElement Exceptions.](#)

Actions Class

Selenium Actions class is used to perform Mouse hover events, keyboard events. User can use this class for advanced actions such as drag and drop, multi selection etc.



The screenshot shows a Java code editor with the file `ActionsClassDemo.java` open. The code demonstrates how to use the Selenium Actions class to perform a search operation. It includes imports for Selenium packages, initializes a ChromeDriver, finds the search input field by name, types "Automation Testing" into it, finds the search button by name, performs a click action on it using the Actions class, and finally closes the browser.

```
1 package selenium.basics;
2 import org.openqa.selenium.By;
3 import org.openqa.selenium.WebDriver;
4 import org.openqa.selenium.WebElement;
5 import org.openqa.selenium.chrome.ChromeDriver;
6 import org.openqa.selenium.interactions.Actions;
7 public class ActionsClassDemo {
8     public static void main(String[] args) throws InterruptedException {
9         // Set Chrome Drive path
10        System.setProperty("webdriver.chrome.driver", "./Resources//chromedriver.exe");
11        // Create driver objet
12        WebDriver driver = new ChromeDriver();
13        // Launch application
14        driver.get("https://www.google.co.in");
15        // Static wait
16        Thread.sleep(3000);
17        // Create WebElement for Search Text box
18        WebElement txtBox = driver.findElement(By.name("q"));
19        // Type Automation Testing in the search Text box
20        txtBox.sendKeys("Automation Testing");
21        // Create WebElement for Google search button
22        WebElement btnSearch = driver.findElement(By.name("btnK"));
23        // Static Wait
24        Thread.sleep(2000);
25        // Create Actions class object
26        Actions actObj = new Actions(driver);
27        // Perform click action on Google Search button using Actions class
28        actObj.moveToElement(btnSearch).click().perform();
29        // Static wait
30        Thread.sleep(2000);
31        // Close Browser tab
32        driver.close();
33        // Quit Browser
34        driver.quit();
35    }
36}
```

Another Example:

```
public class ActionsDemo {  
    public static void main(String[] args) throws InterruptedException {  
        System.setProperty("webdriver.chrome.driver", "./Resources/chromedriver.exe");  
        WebDriver driver = new ChromeDriver();  
        driver.manage().window().maximize();  
        Actions act = new Actions(driver);  
        driver.get("https://www.online.citibank.co.in/");  
        Thread.sleep(2000);  
        WebElement lnkCreditCards = driver.findElement(By.xpath("//a[@title='Credit Cards' and @id='topMnucreditcards']"));  
        act.moveToElement(lnkCreditCards).perform();  
        Thread.sleep(10000);  
        driver.close();  
        driver.quit();  
    }  
}
```

JavascriptExecutor

Sometimes Selenium WebDriver gives ElementNotInteractableException then user might need JavascriptExecutor. Please look at below URL for more information.

<https://www.browserstack.com/guide/javascriptexecutor-in-selenium>

The screenshot shows an IDE window with a Java file named `JavaScriptExecutorDemo.java`. The code imports various Selenium packages and defines a `main` method that sets up a ChromeDriver, navigates to Google, performs a search, and interacts with a search button using `executeScript`. The terminal output on the right shows the driver starting, navigating to Google, performing the search, and interacting with the search button.

```
1 package practice;  
2  
3 import org.openqa.selenium.By;  
4 import org.openqa.selenium.JavascriptExecutor;  
5 import org.openqa.selenium.WebDriver;  
6 import org.openqa.selenium.WebElement;  
7 import org.openqa.selenium.chrome.ChromeDriver;  
8  
9 public class JavaScriptExecutorDemo {  
10  
11     public static void main(String[] args) throws InterruptedException {  
12         System.setProperty("webdriver.chrome.driver", "./Resources/chromedriver.exe");  
13         WebDriver driver = new ChromeDriver();  
14  
15         driver.get("https://www.google.co.in");  
16  
17         driver.findElement(By.xpath("//input[@name='q']")).sendKeys("Automation Testing using BlazeMeter Platform");  
18         Thread.sleep(3000);  
19  
20         WebElement btnSearch = driver.findElement(By.xpath("//div[@jsname='VlcLAE']//input[@name='btnK']"));  
21  
22         JavascriptExecutor executor = (JavascriptExecutor) driver;  
23         executor.executeScript("arguments[0].click()", btnSearch);  
24  
25         driver.close();  
26         driver.quit();  
27     }  
28 }  
29 }
```

```
<terminated> JavaScriptExecutorDemo [Java Application] Starting ChromeDriver 96.0.4664.45 (7c57e0d0d6a8c8438eefef9283a3d60a) on port 5205 Only local connections are allowed. Please see https://chromedriver.chromium.org/ ChromeDriver was started successfully! [1641766748.352][WARNING]: This version is experimental. Jan 10, 2022 3:49:08 AM org.openqa.selenium.remote.ProtocolHandshake INFO: Detected dialect: W3C Jan 10, 2022 3:49:08 AM org.openqa.selenium.remote.HttpCommandExecutor WARNING: Unable to find an exact match for /session/:/value Jan 10, 2022 3:49:08 AM org.openqa.selenium.remote.HttpCommandExecutor INFO: Found CDP implementation for version 96.0.4664.45
```

Scroll Up and Scroll Down using JavascriptExecutor

```
//scroll down  
js.executeScript("window.scrollBy(0,250)", "");  
  
//Scroll up  
js.executeScript("window.scrollBy(0,-350)", "");
```

Move to Element using JavascriptExecutor

```
JavascriptExecutor js = (JavascriptExecutor) driver;  
WebElement lnkHelp = driver.findElement(By.xpath("//a[text()='Help']"));  
js.executeScript("arguments[0].scrollIntoView()", lnkHelp);
```

Working with Edit / Text box

Use `.sendKeys` method to type or enter any text in an Edit or Text box of your webpage.

Approach #1:

```
driver.findElement(By.xpath("//input[@name='q']")).sendKeys("Automation Testing using  
BlazeMeter Platform");
```

Approach #2:

```
WebElement searchBar = driver.findElement(By.xpath("//input[@name='q']"));  
searchBar.sendKeys("Automation Testing using BlazeMeter Platform");
```

Working with check box

Use .click() method to check or uncheck a check box.

Use .isSelected() method to get the check box status that is selected or not.

The screenshot shows the Eclipse IDE interface. The top part displays the code for 'CheckBoxDemo.java'. The bottom part shows the 'Console' tab with the application's log output.

```
CheckBoxDemo.java
3 import org.openqa.selenium.By;
4 public class CheckBoxDemo {
5     public static void main(String[] args) throws InterruptedException {
6         System.setProperty("webdriver.chrome.driver", "./Resources/chromedriver.exe");
7         WebDriver driver = new ChromeDriver();
8         driver.get("https://www.walmart.com/account/login?vid=oao&tid=0&returnUrl=%2F");
9         WebElement chk1 = driver.findElement(By.xpath("//input[@id='remember-me']"));
10        Thread.sleep(2000);
11        ((JavascriptExecutor)driver).executeScript("arguments[0].click();", chk1);
12        Thread.sleep(1000);
13        System.out.println(chk1.isSelected());
14        ((JavascriptExecutor)driver).executeScript("arguments[0].click();", chk1);
15        System.out.println(chk1.isSelected());
16        Thread.sleep(2000);
17        driver.close();
18        driver.quit();
19    }
20 }
```

Console Output:

```
<terminated> CheckBoxDemo [Java Application] C:\Program Files\Java\jdk1.8.0_271\bin\javaw.exe (10-Jan-2022, 4:28:33 am – 4:28:46 am)
ChromeDriver was started successfully.
[1641769116.743][WARNING]: This version of ChromeDriver has not been tested with Chrome version 97.
Jan 10, 2022 4:28:36 AM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
Jan 10, 2022 4:28:37 AM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find an exact match for CDP version 97, so returning the closest version found: 96
Jan 10, 2022 4:28:37 AM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
INFO: Found CDP implementation for version 97 of 96
false
true
```

Working with option button

Use .click() method to select any option button.

Working with Dropdown list box

```
WebElement dropDown = driver.findElement(By.id("Best Match"));
Select select1 = new Select(dropDown);
select1.selectByIndex(2);
select1.selectByValue("Price Low to High");
System.out.println(select1.getAllSelectedOptions());
```

```
System.out.println(select1.getOptions());
System.out.println(select1.getFirstSelectedOption());
```

Refer [this link](#) for more information.

Working with Web Table

There are multiple ways to interact with a Web Table in a Webpage. User might need to use XPath Axes when there is no unique name. Here is an example:

<https://cosmocode.io/automation-practice-webtable/>

//strong[text()='Algeria']//preceding::input[1]

<input type="checkbox"/>	Afghanistan	Kabul	Afghani	Dari Persian; Pashto
<input type="checkbox"/>	Albania	Tirane	Lek	Albanian
<input checked="" type="checkbox"/> hasVisited	Algeria	Algiers	Algerian Dinar	Arabic; Tamazight; French
<input type="checkbox"/>	Andorra	Andorra la Vella	Euro	Catalan
<input type="checkbox"/>	Angola	Luanda	Kwanza	Portuguese

Elements Console Sources Network Performance Memory Application Security Lighthouse Recorder

Styles Computed Layout Event Listeners

Filter :hover .cls + []

element.style { }
b, strong { font-weight: 700; }
a, abbr, acronym, address, applet, big, blockquote, body, caption, cite, code, dd, del, dfn, div, dl, dt, em, strong { font-weight: 700; }

main#main.site-main article#post-2338.post-2338.page-type-page.status-publish div.inside-article div.entry-content table#countries tbody tr td strong ...

//strong[text()='Algeria']//preceding::input[1]

Console What's New

Highlights from the Chrome 97 update

New preview feature: Recorder panel

Working with Frames

iFrame (Inline Frame) is a web page embedded in a web page. Selenium WebDriver could not find any WebElements inside iFrame directly. User need to switch to iFrame to proceed further testing.

```
FramesDemo.java
9
10 public class FramesDemo {
11
12     public static void main(String[] args) throws InterruptedException {
13         System.setProperty("webdriver.chrome.driver", "./Resources//chromedriver.exe");
14         WebDriver driver = new ChromeDriver();
15         driver.manage().window().maximize();
16         driver.get("https://demoqa.com/frames");
17         Thread.sleep(2000);
18
19         driver.switchTo().frame("frame1");
20         List<WebElement> strHeader = driver.findElements(By.xpath("//h1[@id='sampleHeading']"));
21         System.out.println("Total H1 Headers found in frame1: " + strHeader.size());
22         Thread.sleep(2000);
23         driver.close();
24         driver.quit();
25     }
26 }
```

Problems Console @ Javadoc Declaration Search Servers Debug

<terminated> FramesDemo [Java Application] C:\Program Files\Java\jdk1.8.0_271\bin\javaw.exe (12-Jan-2022, 6:43:23 pm – 6:43:41 pm)

ChromeDriver was started successfully.

[1641993207.547][WARNING]: This version of ChromeDriver has not been tested with Chrome version 97.

Jan 12, 2022 6:43:27 PM org.openqa.selenium.remote.ProtocolHandshake createSession

INFO: Detected dialect: W3C

Jan 12, 2022 6:43:27 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch

WARNING: Unable to find an exact match for CDP version 97, so returning the closest version found: 96

Jan 12, 2022 6:43:27 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch

INFO: Found CDP implementation for version 97 of 96

Total H1 Headers found in frame1: 1

Capture Screenshot

```
J ScreenshotDemo.java X
1 package practice;
2
3 import java.io.File;
4 import java.io.IOException;
5
6 import org.apache.commons.io.FileUtils;
7 import org.openqa.selenium.OutputType;
8 import org.openqa.selenium.TakesScreenshot;
9 import org.openqa.selenium.WebDriver;
10 import org.openqa.selenium.chrome.ChromeDriver;
11
12 public class ScreenshotDemo {
13
14     public static void main(String[] args) throws IOException {
15         System.setProperty("webdriver.chrome.driver", "./Resources//chromedriver.exe");
16         String fileName = "./Resources//screenshot1.png";
17         WebDriver driver = new ChromeDriver();
18         driver.manage().window().maximize();
19         driver.get("https://www.citi.com");
20         // Convert web driver object to TakeScreenshot
21         TakesScreenshot scrShot = ((TakesScreenshot) driver);
22         // Call getScreenshotAs method to create image file
23         File SrcFile = scrShot.getScreenshotAs(OutputType.FILE);
24         // Move image file to new destination
25         File DestFile = new File(fileName);
26         // Copy file at destination
27         FileUtils.copyFile(SrcFile, DestFile);
28
29         driver.close();
30         driver.quit();
31     }
}
```

Navigation Commands

```
driver.navigate().to("www.google.com");
driver.navigate().forward();
driver.navigate().back();
driver.navigate().refresh();
```

Alerts

Frequently used alerts methods in test automation.

```
Alert alert = driver.switchTo().alert();
alert.dismiss();
alert.getText();
alert.accept();
alert.sendKeys("Hello World.");
```

Selenium Wait

Selenium WebDriver provides three commands to implement waits in tests.

1. Implicit Wait

```
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS); //Sel <= 4
```

```
driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(10)); //Sel >4.x
```

2. Explicit Wait

```
WebDriverWait explicitWait = new WebDriverWait(driver, 20);
```

```
//Selenium 4 syntax  
 WebDriverWait wait = new WebDriverWait(driver,Duration.ofSeconds(10));  
 wait.until(ExpectedConditions.visibilityOfElementLocated(By.cssSelector(".classlocator")));
```

3. Fluent Wait

Before Selenium 4 -

```
Wait<WebDriver> wait = new FluentWait<WebDriver>(driver)  
 .withTimeout(30, TimeUnit.SECONDS)  
 .pollingEvery(5, TimeUnit.SECONDS)  
 .ignoring(NoSuchElementException.class);
```

After Selenium 4 -

```
Wait<WebDriver> fluentWait = new FluentWait<WebDriver>(driver)  
 .withTimeout(Duration.ofSeconds(30))  
 .pollingEvery(Duration.ofSeconds(5))  
 .ignoring(NoSuchElementException.class);
```

<https://devqa.io/webdriver-explicit-implicit-fluent-wait/>

Browser Driver Options

Before Selenium 4 there was DesiredCapabilities class exists and we used these capabilities to define test environment such as OS, Browser name, version etc. In Selenium 4.x replaced with options object.

- Chrome – ChromeOptions
- Microsoft Edge – EdgeOptions
- Firefox – FirefoxOptions
- Safari – SafariOptions
- Internet Explorer (IE) – InternetExplorerOptions

```

3@ import java.util.Arrays;
4 import java.util.HashMap;
5 import java.util.Map;
6
7 import org.openqa.selenium.chrome.ChromeDriver;
8 import org.openqa.selenium.chrome.ChromeOptions;
9
10 public class ChromeOptionsDemo {
11
12@ public static void main(String[] args) throws InterruptedException {
13     System.setProperty("webdriver.chrome.driver", "./Resources/chromedriver.exe");
14
15     ChromeOptions options = new ChromeOptions();
16     //Open Browser in maximized mode
17     options.addArguments("start-maximized");
18     //Block popups
19     options.setExperimentalOption("excludeSwitches",Arrays.asList("disable-popup-blocking"));
20
21     //Set default download directory
22     Map<String, Object> prefs = new HashMap<String, Object>();
23     prefs.put("download.default_directory", "/directory/path");
24     options.setExperimentalOption("prefs", prefs);
25
26     ChromeDriver driver = new ChromeDriver(options);
27
28     driver.get("https://www.google.co.in");
29     Thread.sleep(2000);
30     driver.close();
31     driver.quit();
32 }
33 }
```

Refer [this link](#) for more browser options.

Page Object Model

<https://www.toolsqa.com/selenium-webdriver/page-object-model/>

Page Factory Design Pattern

<https://www.browserstack.com/guide/page-object-model-in-selenium>

@FindBy to be used while initializing WebElement.

User need to use PageFactory.initElement(driver, this.class) to be used in the page class constructor.

Maven

Build automation tool, download dependencies automatically. There are many other build automation tools available. Example: Gradle, Ant etc.

Maven Lifecycle Phases

<https://maven.apache.org/guides/introduction/introduction-to-the-lifecycle.html#a-build-lifecycle-is-made-up-of-phases>

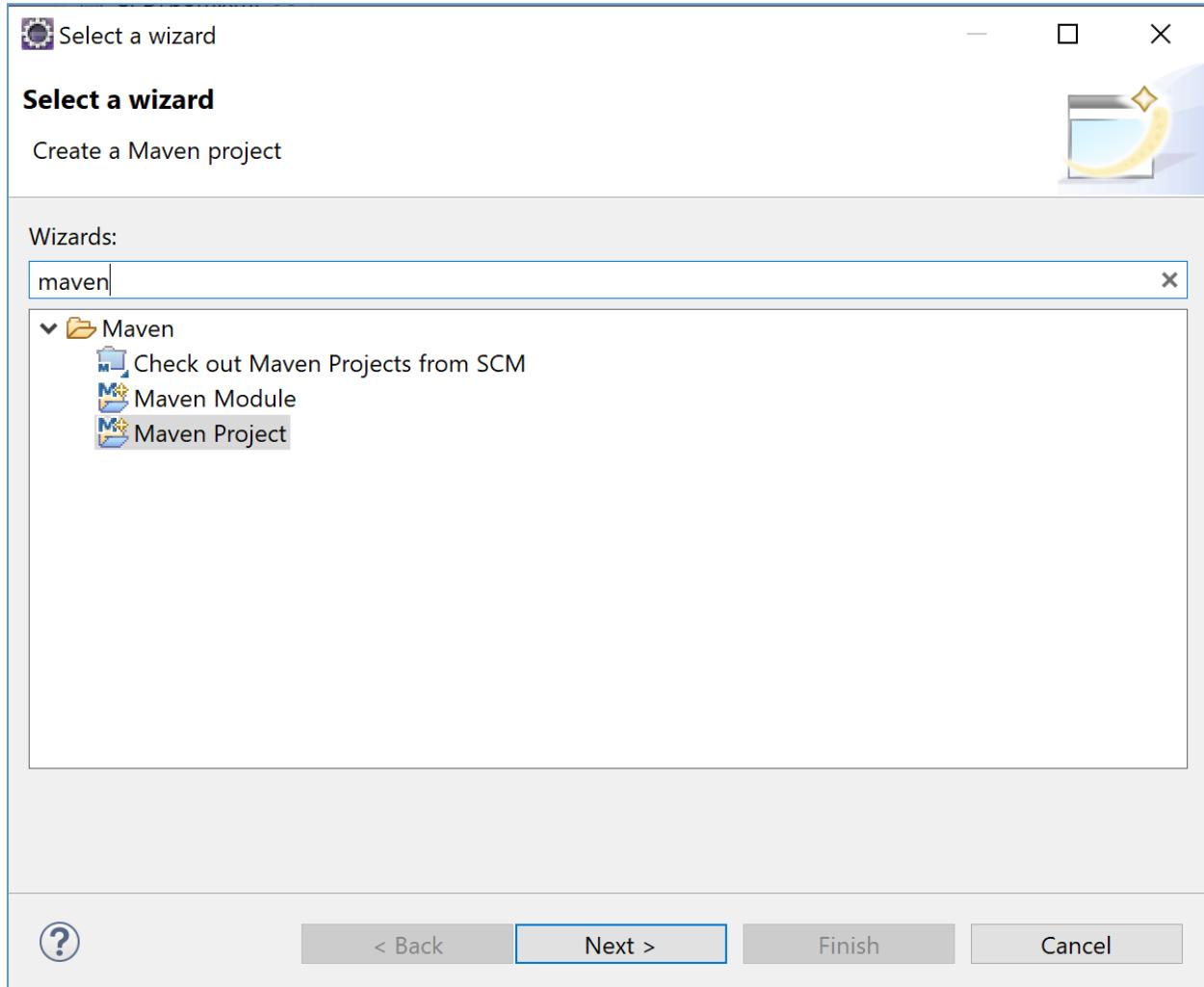
Plug-ins vs Dependencies

Plugin is a Jar file which executes the task, and dependency is a Jar which provides the class files to execute the task.

Important points in Maven

1. Maven downloads all dependencies JARs in user profile .m2 folder. Example: C:\users\stata\.m2
2. User must resolve all POM issues before build the Maven project.
3. Maven POM means Project Object Model
4. Maven commands can be executed from CLI (Command Line Interface) too.

Step 1:



Step 2:

New Maven Project

New Maven project

Select project name and location

Create a simple project (skip archetype selection)

Use default Workspace location

Location:

Add project(s) to working set

Working set:

► Advanced

Step 3:

New Maven Project

New Maven project

Configure project

Artifact

Group Id: EmpApps

Artifact Id: CPD1

Version: 0.0.1-SNAPSHOT

Packaging: jar

Name:

Description:

Parent Project

Group Id:

Artifact Id:

Version:

► Advanced

WebDriverManager

WebDriverManager is an open-source Java library that carries out the management (i.e., download, setup, and maintenance) of the drivers required by Selenium WebDriver (e.g., chromedriver, geckodriver, msedgedriver, etc.) in a fully automated manner.

WebDriverManager Maven Dependency

<https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager>

Selenium Server Maven Dependency

<https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-server>

TestNG

Test Next Generation Framework, users can use TestNG for test suite execution, batch, group and parallel execution purposes. Also, it has reporting capability.

TestNG Maven Dependency

<https://mvnrepository.com/artifact/org.testng/testng>

TestNG Annotations

1. @BeforeSuite
2. @BeforeTest
3. @BeforeMethod
4. @Test
5. @AfterMethod
6. @AfterTest
7. @AfterSuite

TestNG Reports

Use Reporter.log to add custom message in the TestNG Report. Examples:

```
Reporter.log("Google search for Iphone 13 done successfully");
```

Attach Screenshot

```
Reporter.log("<a href='" + destFile.getAbsolutePath() + "'> <img src='" +  
destFile.getAbsolutePath() + "' height='100' width='100'> </a>");
```



TestNG_Report_Screenshot_Example.txt

TestNG Assertions

TestNG provides various assertions to validate actual results against expected results. Few examples:

1. assertTrue
 - a. Assert.assertTrue(condition)
2. assertFalse
 - a. Assert.assertFalse(condition)
3. assertEquals
 - a. Assert.assertEquals(actual, expected)
4. assertNotEquals
 - a. Assert.assertNotEquals(actual, expected, message)

TestNG Listeners

Listeners are TestNG annotations that literally “listen” to the events in a script and modify TestNG behaviour accordingly. These listeners are applied as interfaces in the code. For example, the most common usage of listeners occurs when taking a screenshot of a particular test that has failed along with the reason for its failure. Listeners also help with logging and generating results.

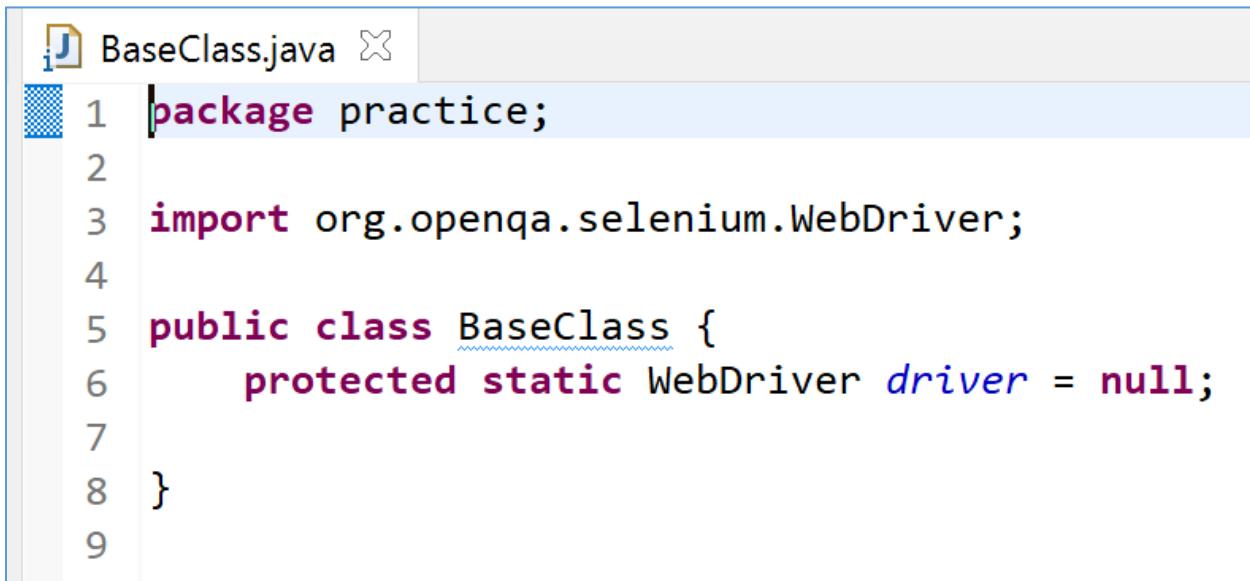
1. IAnnotationTransformer
2. IAnnotationTransformer2
3. IConfigurable
4. IConfigurationListener
5. IExecutionListener

6. IHookable
7. IInvokedMethodListener
8. IInvokedMethodListener2
9. IMethodInterceptor
10. IReporter
11. ISuiteListener
12. ITestListener

Listener Example:

1. Create a BaseClass with WebDriver object.
2. Create a Java class to automate manual test cases, extend base class to get driver object.
3. Create another Java class for listener (**extend Base class and implement ITestListener to capture screenshots automatically for every failure and attach to Reporter Log**)
4. Add java class listener in TestNG suite xml.
 - a. User can also add listeners to java classes.

Step 1 code:



The screenshot shows a Java code editor with a file named "BaseClass.java". The code defines a public class "BaseClass" with a protected static WebDriver variable named "driver" initialized to null. The code is numbered from 1 to 9 on the left side.

```
1 package practice;
2
3 import org.openqa.selenium.WebDriver;
4
5 public class BaseClass {
6     protected static WebDriver driver = null;
7
8 }
9
```

Step 2 code:

```
DriverExample.java ✘
1 package practice;
2
3④ import java.io.File;□
4
5 public class DriverExample extends BaseClass{
6     @BeforeTest()
7     public void setup() {
8         WebDriverManager.chromedriver().setup();
9         driver = new ChromeDriver();
10    }
11    @Test(priority = 1)
12    public void performGoogleSearch() {
13        driver.get("https://www.google.com");
14        driver.findElement(By.xpath("//input[@name='q']")).sendKeys("Apple Iphone 13" + Keys.ENTER);
15        Assert.assertTrue(driver.getTitle().contains("Apple Iphone 14"));
16        Reporter.Log("Google search for Iphone 13 done successfully");
17    }
18    @Test(priority = 2)
19    public void performWikiSearch() {
20        driver.get("https://en.wikipedia.org/wiki/Main_Page");
21        driver.findElement(By.xpath("//input[@name='search']")).sendKeys("iphone 13" + Keys.ENTER);
22        Reporter.Log("Wiki search for Iphone 13 done successfully");
23    }
24    @AfterTest
25    public void tearDown() {
26        driver.close();
27        driver.quit();
28    }
29}
30
```

Step 3 code:

```
ScreenshotListener.java ✘
1 package practice;
2
3④ import java.io.File;□
4
5 public class ScreenshotListener extends BaseClass implements ITestListener {
6
7    public void onTestFailure(ITestResult result) {
8        DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyyMMddHHmmss");
9        LocalDateTime now = LocalDateTime.now();
10       File scrFile = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);
11       try {
12           String reportDirectory = new File(System.getProperty("user.dir")).getAbsolutePath()
13               + "/target/surefire-reports";
14           File destFile = new File(reportDirectory + "/failure_screenshots/screenshot_" + dtf.format(now) + ".png");
15           FileUtils.copyFile(scrFile, destFile);
16           Reporter.Log("<a href='" + destFile.getAbsolutePath() + "'> <img src='" + destFile.getAbsolutePath()
17               + "' height='100' width='100'> </a>");
18       } catch (IOException e) {
19           e.printStackTrace();
20       }
21   }
22}
23
```

Step 4 Code:

```
ListenerSuite.xml
1 <!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
2
3<suite name="Suite1">
4    <listeners>
5        <listener class-name="practice.ScreenshotListener" />
6    </listeners>
7    <test name="Regression">
8        <classes>
9            <class name="practice.DriverExample" />
10       </classes>
11    </test>
12 </suite>
```

Add Listener only to a specific Java class then use @Listeners annotation. Example:

```
import org.testng.annotations.Listeners;
import org.testng.annotations.Test;
import io.github.bonigarcia.wdm.WebDriverManager;
@Listeners(practice.ScreenshotListener.class)
public class DriverExample extends BaseClass{
```

TestNG Suite XML

Sample TestNG Suite XML:

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >

<suite name="Suite1" verbose="1" >
    <test name="Regression" >
        <classes>
            <class name="practice.DriverExample" />
        </classes>
    </test>
</suite>
```

Refer [this link](#) for more TestNG Suite XML options.

TestNG Parallel Execution

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
<suite name = "Parallel Testing Suite">
    <test name = "Parallel Tests" parallel = "methods">
        <classes>
            <class name = "ParallelTest" />
        </classes>
```

```
</test>  
</suite>
```

Maven TestNG Integration – SureFirePlugin

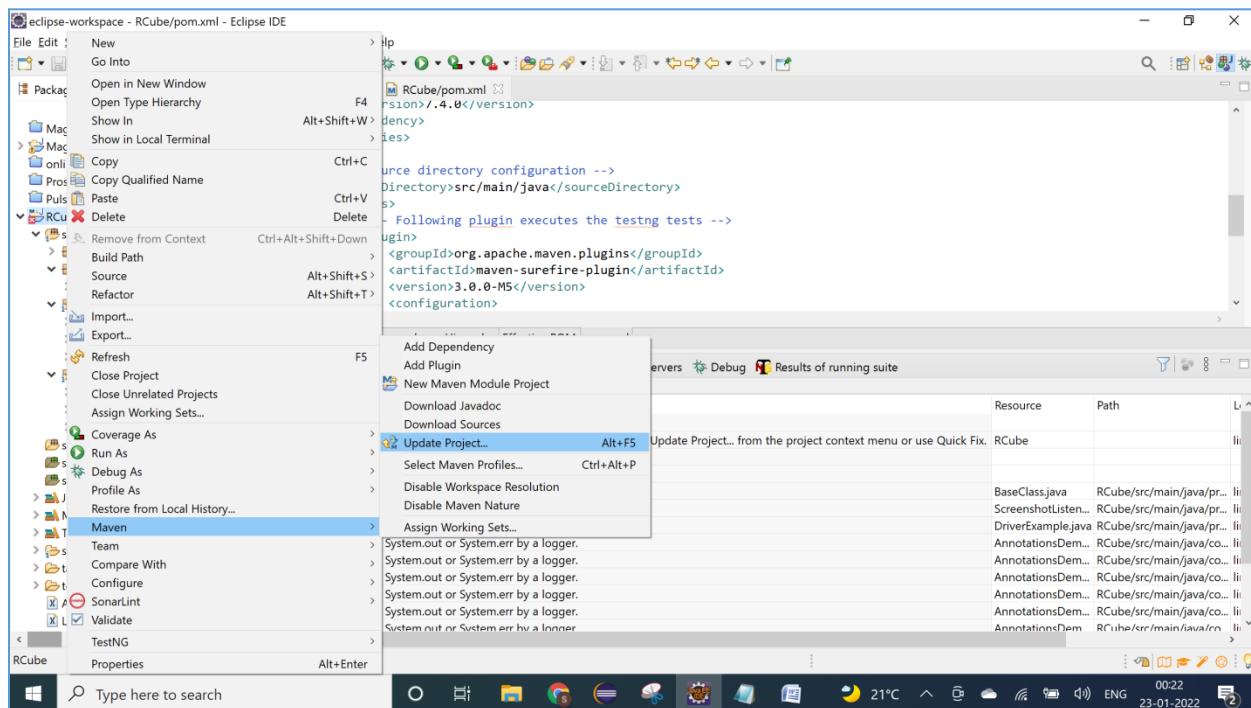
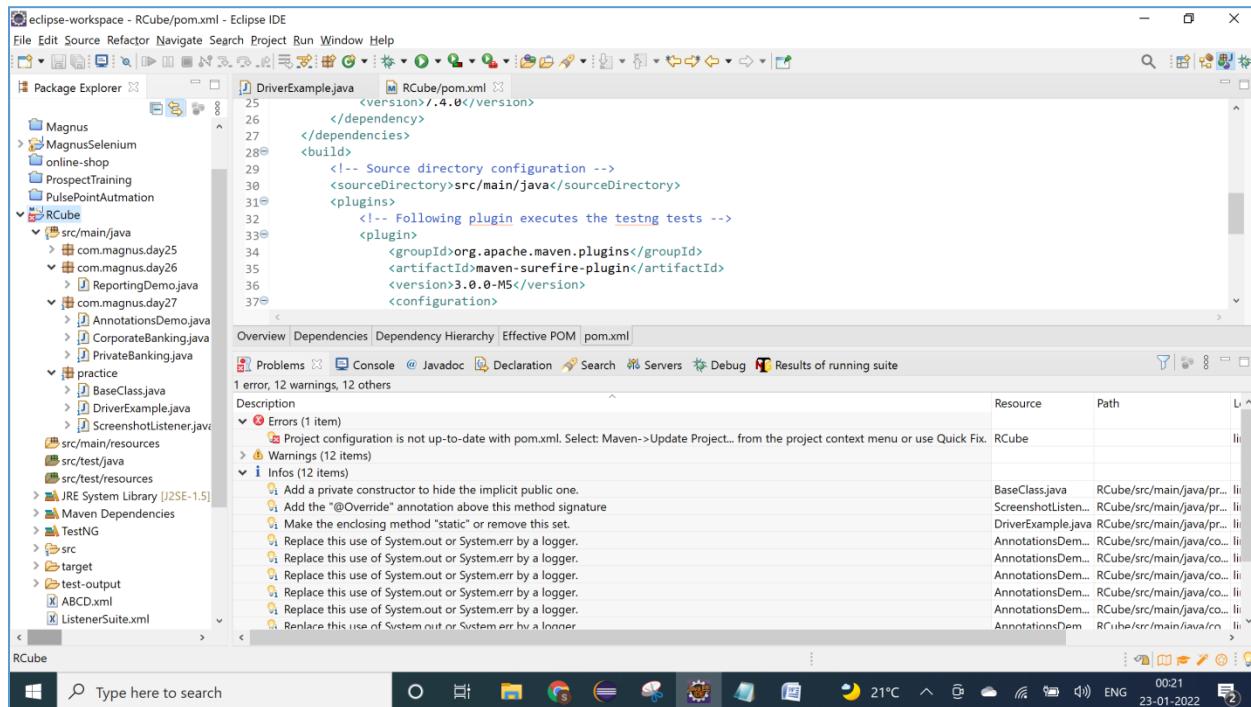
To execute TestNG Suite from Maven then you need SureFirePlugin in the Maven POM.xml.

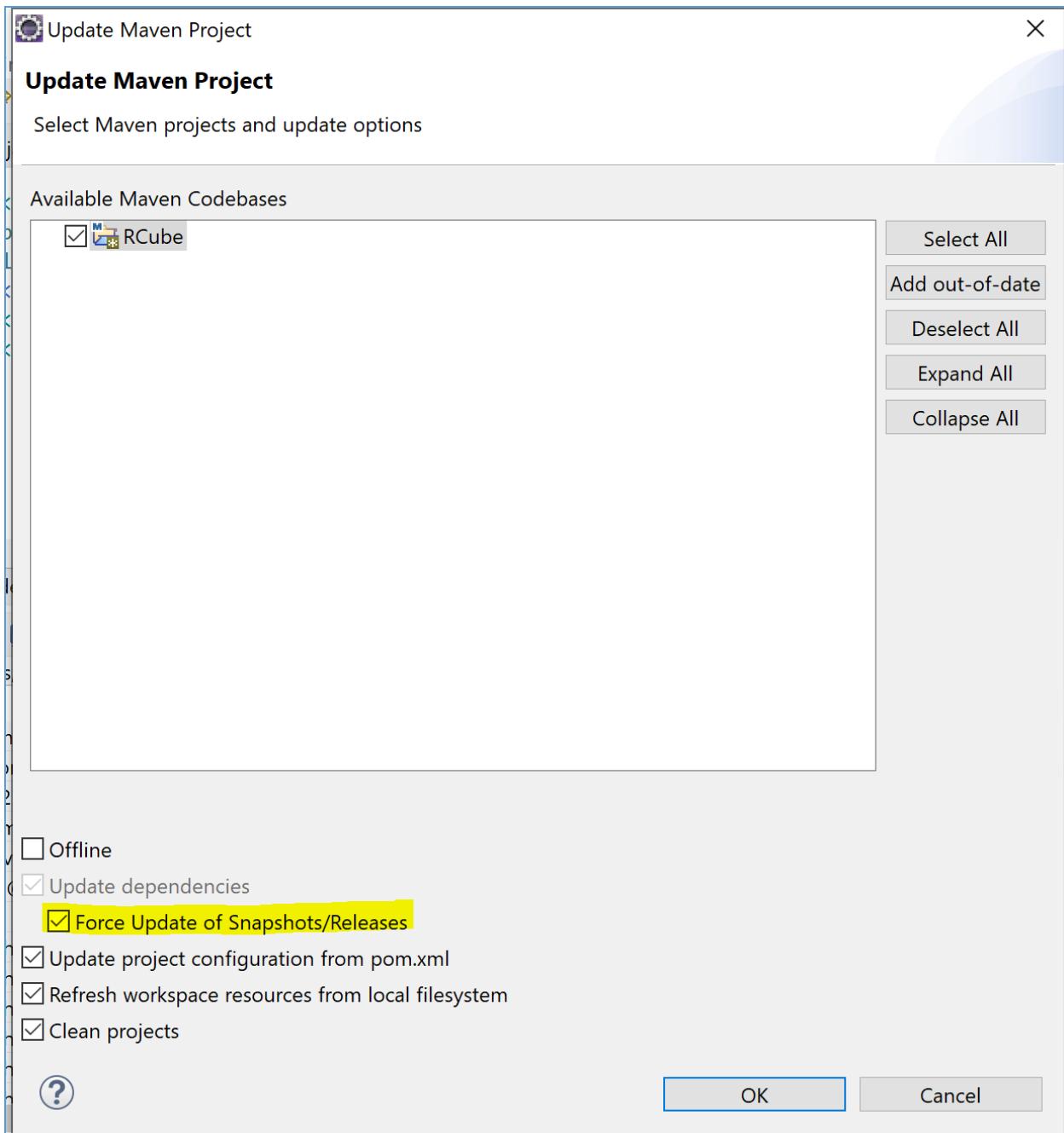
```
27      </dependencies>  
28  <build>  
29      <!-- Source directory configuration -->  
30      <sourceDirectory>src/main/java</sourceDirectory>  
31  <plugins>  
32      <!-- Following plugin executes the testng tests -->  
33  <plugin>  
34      <groupId>org.apache.maven.plugins</groupId>  
35      <artifactId>maven-surefire-plugin</artifactId>  
36      <version>3.0.0-M5</version>  
37  <configuration>  
38      <!-- Suite testing xml file to consider for test execution -->  
39  <suiteXmlFiles>  
40      <suiteXmlFile>ABCD.xml</suiteXmlFile>  
41      <suiteXmlFile>RegressionTests.xml</suiteXmlFile>  
42  </suiteXmlFiles>  
43  </configuration>  
44  </plugin>  
45  <!-- Compiler plugin configures the java version to be used for compiling the code -->  
46  <plugin>  
47      <artifactId>maven-compiler-plugin</artifactId>  
48  <configuration>  
49      <source>1.8</source>  
50      <target>1.8</target>  
51  </configuration>  
52  </plugin>  
53  </plugins>  
54  </build>  
55 </project>
```

Maven Compiler Plugin and Surefire plugin Code:

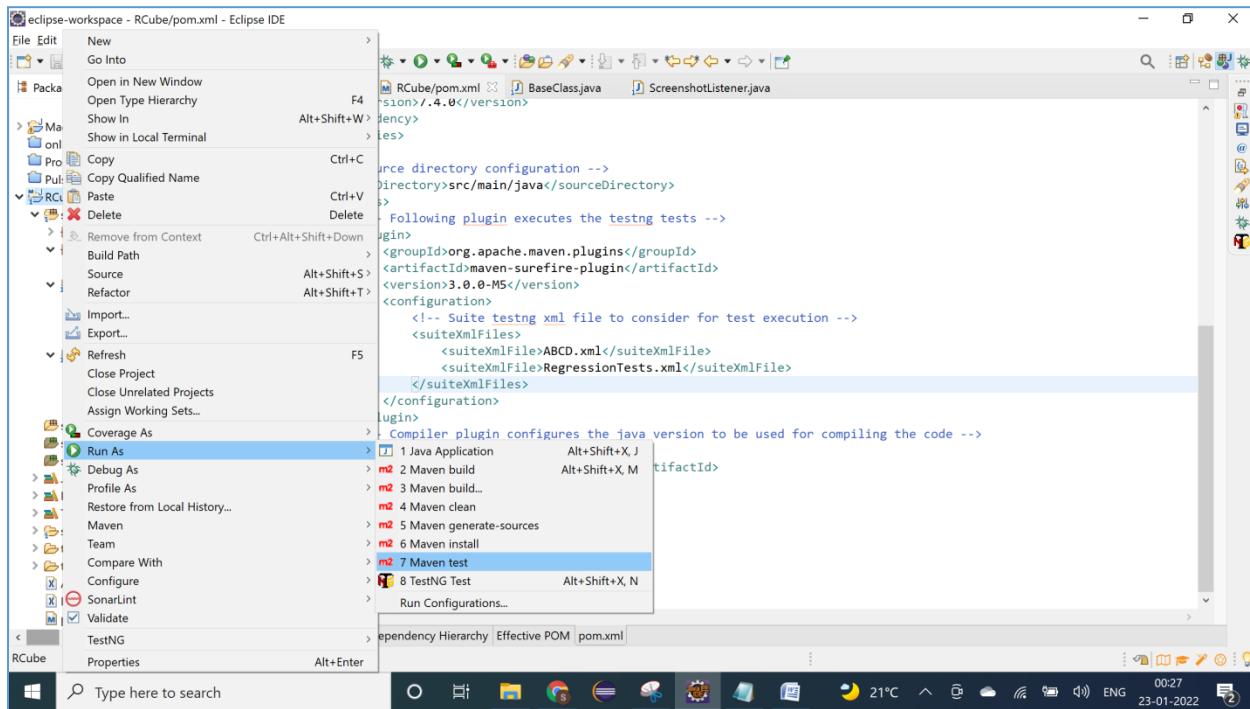
```
<build>  
    <!-- Source directory configuration -->  
    <sourceDirectory>src/main/java</sourceDirectory>  
    <plugins>  
        <!-- Following plugin executes the testing tests -->  
        <plugin>  
            <groupId>org.apache.maven.plugins</groupId>  
            <artifactId>maven-surefire-plugin</artifactId>  
            <version>3.0.0-M5</version>  
            <configuration>  
                <!-- Suite testng xml file to consider for test execution -->  
                <suiteXmlFiles>  
                    <suiteXmlFile>ABCD.xml</suiteXmlFile>  
                    <suiteXmlFile>RegressionTests.xml</suiteXmlFile>  
                </suiteXmlFiles>  
            </configuration>  
        </plugin>  
        <!-- Compiler plugin configures the java version to be used for compiling the code -->  
        <plugin>  
            <artifactId>maven-compiler-plugin</artifactId>  
            <configuration>  
                <source>1.8</source>  
                <target>1.8</target>  
            </configuration>  
        </plugin>  
    </plugins>  
</build>
```

Maven Force Update Project





Run Maven Test



GIT

It is distributed version controlling system, users keep their dev and test related code in GIT.

Pre-requisites

1. Create an account in Github
2. Create a repository in Github
3. Install Git Client which gives GIT GUI and GIT Bash
4. Create GIT personal access token for authentication to be used in GIT BASH for code pull and push.

GIT Bash

It is a command line interface to push or pull code into Github.

Example of Git Repo: <https://github.com/subramanyam7/Batch11.git>

You can create a GIT Personal Access Token for your account to pull and push code.

Refer [this video](#) for steps.

1. git clone <repo> (**You can clone any public repo**)
 - a. git clone <https://<YourUserID>:PAT@github.com/subramanyam7/Batch11.git> (**You can clone a public repo and make changes to it after cloning and push back to github if you are a Collaborator to that repo**)
2. git add .
3. git commit -m "commit message"

4. git push
5. git diff
6. git reset --hard HEAD
7. git clean
8. git pull

Code / Steps push code into Github from Git Bash

```
srinivas@LAPTOP-5I3N02GE MINGW64 ~/Documents/Git
$ git clone https://sudharanik1982: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@gith
ub.com/subramanyam7/Batch11.git
Cloning into 'Batch11'...
remote: Enumerating objects: 39, done.
remote: Counting objects: 100% (39/39), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 39 (delta 6), reused 30 (delta 4), pack-reused 0
Unpacking objects: 100% (39/39), done.

srinivas@LAPTOP-5I3N02GE MINGW64 ~/Documents/Git
$ cd Batch11/

srinivas@LAPTOP-5I3N02GE MINGW64 ~/Documents/Git/Batch11 (main)
$ git add .

srinivas@LAPTOP-5I3N02GE MINGW64 ~/Documents/Git/Batch11 (main)
$ git commit -m "added a comment in regression tests xml"
[main 1aee54f] added a comment in regression tests xml
 2 files changed, 11 insertions(+)
  create mode 100644 RegressionTests.xml.bak

srinivas@LAPTOP-5I3N02GE MINGW64 ~/Documents/Git/Batch11 (main)
$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 348 bytes | 348.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/subramanyam7/Batch11.git
  8a5d6a2..1aee54f  main -> main

srinivas@LAPTOP-5I3N02GE MINGW64 ~/Documents/Git/Batch11 (main)
$
```

Note:

If you get any error while Code Push / Pull then try below command and retry code push / pull.

```
dell@DESKTOP-MV6L4GQ MINGW64 ~/Documents/GitDemo/Batch11 (main)
$ git config --global user.email "subramanyam7@yahoo.com"

dell@DESKTOP-MV6L4GQ MINGW64 ~/Documents/GitDemo/Batch11 (main)
$ |
```

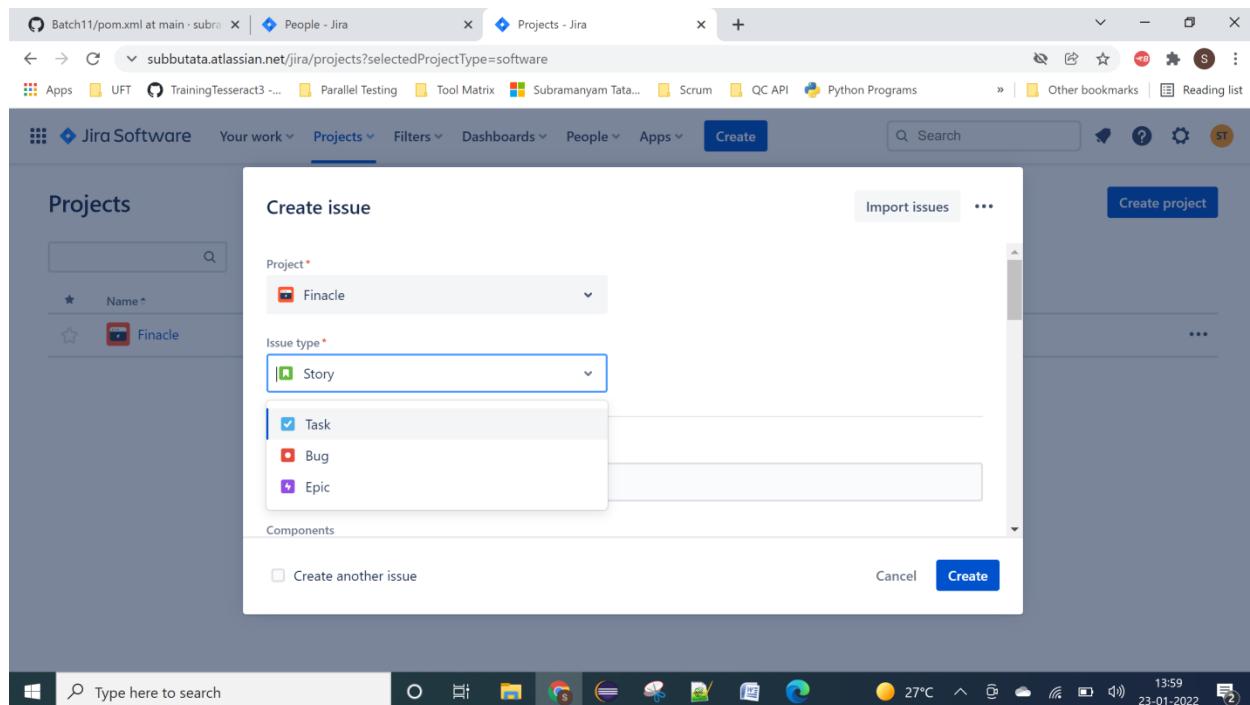
Jira

Project management tool, to track all user stories, sub-tasks, defects in a dashboard.

[Click here](#) to get JIRA free trial.

Issue types in JIRA:

- Story
- Task
- Bug
- Epic



BDD

Behavior Driven Development, also called Business Driven Development, we use Cucumber to implement test automation scripts in Gherkin language (Given, When, Then etc.).

1. Define the business requirement as a test (feature file).
2. Create code to implement step features
3. Create runner class to execute multiple features

Pre-requisites

1. Install Cucumber plugin from Eclipse Marketplace
2. Install Natural plugin from Eclipse Help Menu > Install new software

Eclipse Marketplace

Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.

Search Recent Popular Favorites Installed Giving IoT an Edge

Find: X All Markets ▼ All Categories ▼ Go

Cucumber Eclipse Plugin 1.0.0.202106240526

An Eclipse plugin for Cucumber . New Feature list in the version available from the update site: Lambda Expression support for Cucumber-Java8 Content Assistance... [more info](#)

by [Cucumber](#), MIT
[cucumber](#) [Gherkin](#) [JVM](#)[cucumber Plugins ATDD](#)

247 Installs: **422K** (14,014 last month) Install

Natural 0.9

Natural is a set of plugins providing a smooth experience in editing and maintaining BDD/ATDD files. The currently supported languages are Cucumber (Gherkin... [more info](#)

by [rlogiacco@gmail.com](#) Lo Giacco, EPL
[cucumber](#) [Gherkin](#) [JBehave](#) [ATDD](#) [BDD](#)

67 Installs: **266K** (0 last month) Install

If you are unable to install Natural 0.9 from Eclipse Marketplace then try with Help > Install New Software and the URL for Natural plugin is: <http://rlogiacco.github.io/Natural>

Install

Available Software

Check the items that you wish to install.

Work with: [Natural - http://rlogiacco.github.io/Natural](http://rlogiacco.github.io/Natural) Add... Manage...

Select All Deselect All

Name	Version
> <input checked="" type="checkbox"/> Natural	

2 items selected

Details

Dependencies for Cucumber-Selenium:

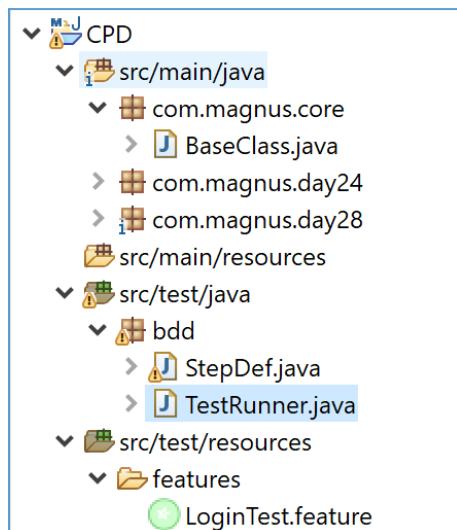
1. Selenium Server
2. Cucumber-jvm
3. JUnit
4. Cucumber-core
5. Cucumber-java
6. Cucumber-junit

Note: Make sure the versions on Cucumber-java, Cucumber -junit and Cucumber-core are the same, i.e., if you are using Cucumber-java-1.2.5 make sure the versions of the other two dependencies are the same.

Cucumber Dependencies POM

```
<dependencies>
    <!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-server -->
    <dependency>
        <groupId>org.seleniumhq.selenium</groupId>
        <artifactId>selenium-server</artifactId>
        <version>4.0.0-alpha-2</version>
    </dependency>
    <!-- https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager -->
    <dependency>
        <groupId>io.github.bonigarcia</groupId>
        <artifactId>webdrivermanager</artifactId>
        <version>5.0.3</version>
    </dependency>
    <dependency>
        <groupId>info.cukes</groupId>
        <artifactId>cucumber-java</artifactId>
        <version>1.2.5</version>
    </dependency>
    <dependency>
        <groupId>info.cukes</groupId>
        <artifactId>cucumber-jvm</artifactId>
        <version>1.2.5</version>
        <type>pom</type>
    </dependency>
    <dependency>
        <groupId>info.cukes</groupId>
        <artifactId>cucumber-core</artifactId>
        <version>1.2.5</version>
    </dependency>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.8.1</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>info.cukes</groupId>
        <artifactId>cucumber-junit</artifactId>
        <version>1.2.5</version>
        <scope>test</scope>
    </dependency>
</dependencies>
```

Selenium Cucumber Project Explorer



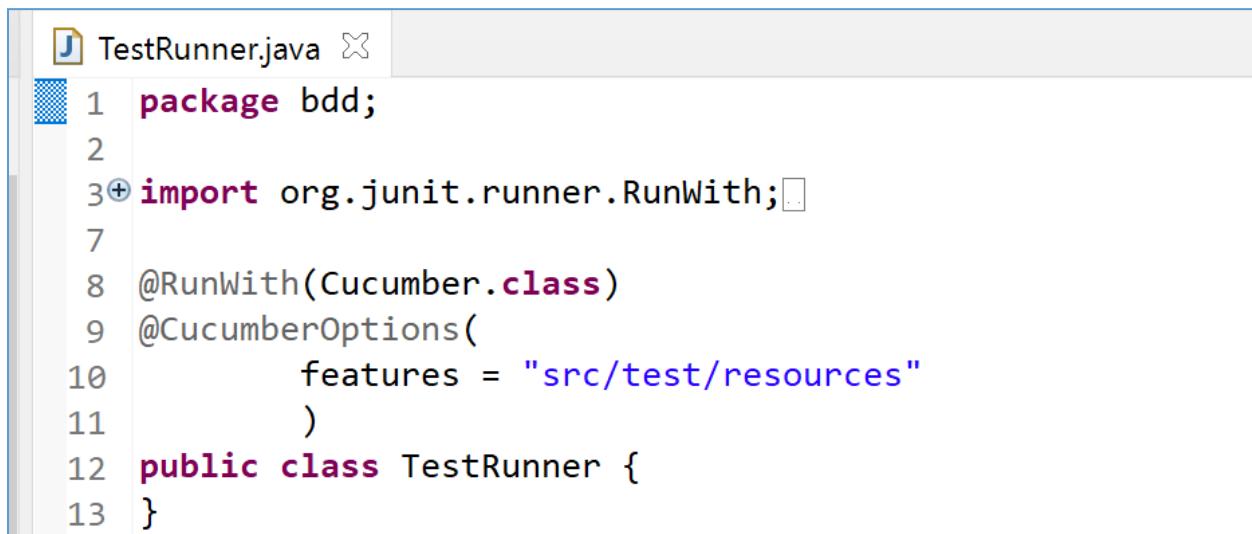
Feature File

```
1 Feature: Login Action
2
3@ Scenario: Perform google search
4@   Given User is on Google Home Page
5@   When User search for UC Deploy
6@   Then UCD search results displayed Successfully|
```

Step Definition File

```
1 package bdd;
2
3@ import java.util.concurrent.TimeUnit;
4
5 public class StepDef extends BaseClass{
6
7     @Given("^User is on Google Home Page$")
8     public void openGoogleHomePage() throws Throwable {
9         WebDriverManager.chromedriver().setup();
10        driver = new ChromeDriver();
11        driver.manage().window().maximize();
12        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
13        driver.get("https://www.google.com/");
14    }
15
16    @When("^User search for UC Deploy$")
17    public void searchUCD() throws Throwable{
18        driver.findElement(By.xpath("//input[@name='q']")).sendKeys("UC Deploy");
19        driver.findElement(By.xpath("//input[@name='q']")).sendKeys(Keys.ENTER);
20    }
21
22    @Then("^UCD search results displayed Successfully$")
23    public void validateResults() throws Throwable{
24        Assert.assertTrue(driver.findElement(By.xpath("//*[contains(text(),'UrbanCode Deploy')]")).isDisplayed());
25        driver.close();
26        driver.quit();
27    }
28}
```

Test Runner File



The screenshot shows a Java code editor window with a tab labeled "TestRunner.java". The code is as follows:

```
1 package bdd;
2
3+import org.junit.runner.RunWith;
4
5 @RunWith(Cucumber.class)
6 @CucumberOptions(
7     features = "src/test/resources"
8 )
9 public class TestRunner {
10 }
```

SQL

Structured Query Language. SQL Command major categories are:

DDL – Data Definition Language

Create, Drop, Alter

DML – Data Manipulation Language

Insert, Delete, Update, Select

DCL – Data Control Language

Grant, Revoke

Refer [this link](#) for more information. Also, use [this link](#) for practice purpose.

API Automation

User can use httpclient dependency to do API automation. If you want test APIs manually then use [Postman](#) or [SoapUI](#) tools.

HttpClient Maven Dependency

```
<dependency>
    <groupId>org.apache.httpcomponents</groupId>
    <artifactId>httpclient</artifactId>
    <version>4.5.12</version>
</dependency>
<dependency>
    <groupId>org.json</groupId>
    <artifactId>json</artifactId>
    <version>20201115</version>
</dependency>
```

Note: You also need JSON dependency to convert response string into JSON object to retrieve data easily.

Implementation

```
package practice;
import java.io.IOException;
import org.apache.http.HttpEntity;
import org.apache.http.client.methods.CloseableHttpResponse;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;
import org.apache.http.util.EntityUtils;
import org.json.JSONObject;

public class AutomateAPI {
    public static void main(String[] args) throws IOException {
        String endPoint = "https://jsonplaceholder.typicode.com/todos/1";
        HttpGet request = new HttpGet(endPoint);
        CloseableHttpClient client = HttpClients.createDefault();
        CloseableHttpResponse response = client.execute(request);
        HttpEntity entity = response.getEntity();
        String result = EntityUtils.toString(entity);
        System.out.println(result);

        JSONObject json = new JSONObject(result);
        System.out.println(json.toString());
        String titleValue = json.getString("title");
        System.out.println(titleValue);
    }
}
```

Output

The screenshot shows an IDE interface with the following details:

- Code Editor:** The file `AutomateAPI.java` is open, containing Java code to send a GET request to `https://jsonplaceholder.typicode.com/todos/1`, parse the JSON response, and print the title.
- Console Output:** The output window shows the JSON response from the API and the printed title value.
- Toolbars:** Standard Eclipse-style toolbars for Problems, Console, Javadoc, Declaration, Search, Servers, Debug, and TestNG.
- Status Bar:** Shows the application is terminated, the file name is `AutomateAPI [Java Application]`, the path is `C:\Program Files\Java\jdk1.8.0_271\bin\javaw.exe`, and the date is `(25-Jan-2022)`.

```
1 package practice;
2
3+ import java.io.IOException;
4 public class AutomateAPI {
5     public static void main(String[] args) throws IOException {
6         String endPoint = "https://jsonplaceholder.typicode.com/todos/1";
7         HttpGet request = new HttpGet(endPoint);
8         CloseableHttpClient client = HttpClients.createDefault();
9         CloseableHttpResponse response = client.execute(request);
10        HttpEntity entity = response.getEntity();
11        String result = EntityUtils.toString(entity);
12        System.out.println(result);
13
14        JSONObject json = new JSONObject(result);
15        System.out.println(json.toString());
16        String titleValue = json.getString("title");
17        System.out.println(titleValue);
18    }
19 }
```

```
{ "userId": 1,
  "id": 1,
  "title": "delectus aut autem",
  "completed": false
}
{"id":1,"completed":false,"title":"delectus aut autem","userId":1}
delectus aut autem
```

SLF4J (Log4J)

Dependencies

```
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-api</artifactId>
    <version>1.7.5</version>
</dependency>
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-log4j12</artifactId>
    <version>1.7.5</version>
</dependency>
```

Implementation

```
import org.apache.log4j.BasicConfigurator;
import org.apache.log4j.Logger;
import org.openqa.selenium.WebDriver;

public class BaseClass {
    static Logger logger = Logger.getLogger("Test Automation");
    public BaseClass() {
        BasicConfigurator.configure();
        logger.info("Entering application.\t" + this.getClass().getName());
    }
    protected static WebDriver driver = null;
}
```

Output

```
[RemoteTestNG] detected TestNG version 7.4.0
0 [main] INFO Test Automation - Entering application. practice.DriverExample
699 [main] DEBUG io.github.bonigarcia.wdm.cache.ResolutionCache - Resolution chrome=97 in cache (valid until 11:09:47 01/02
701 [main] DEBUG io.github.bonigarcia.wdm.cache.ResolutionCache - Resolution chrome97=97.0.4692.71 in cache (valid until 18
701 [main] INFO io.github.bonigarcia.wdm.WebDriverManager - Using chromedriver 97.0.4692.71 (resolved driver for Chrome 97)
719 [main] DEBUG io.github.bonigarcia.wdm.WebDriverManager - Driver chromedriver 97.0.4692.71 found in cache
721 [main] INFO io.github.bonigarcia.wdm.WebDriverManager - Exporting webdriver.chrome.driver as C:\Users\dell\.cache\selen
Starting ChromeDriver 97.0.4692.71 (adefaa7837d02a07a604c1e6efffb3a09422ab88d-refs/branch-heads/4692@{#1247}) on port 47127
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
Feb 01, 2022 10:50:55 AM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
2502 [main] INFO Test Automation - Entered in Google Search method. practice.DriverExample
PASSED: performGoogleSearch
PASSED: performWikiSearch
```