



Selenium

Class 1

Agenda

What is Selenium? History of the Selenium Project

Selenium Components / Selenium's Tools suite

Platforms supported by Selenium

Advantages of Selenium and Disadvantages of Selenium

Testing Frameworks and other tools used in Selenium

First Selenium Webdriver script

Locators

Selenium

Selenium is a free (open source) automated testing suite for web applications across different browsers and platforms

Selenium is not just a single tool but a suite of software, each catering to different testing needs of an organization.

Selenium WebDriver tool is used to automate web application testing to verify that it works as expected.

Selenium

Selenium WebDriver supports many browsers such as Firefox, Chrome, IE, and Safari.

Using the Selenium WebDriver, we can automate testing for **web applications only**. It does not qualify for window-based applications.

Selenium WebDriver supports different programming languages such as C#, Java, Perl, PHP and Ruby for writing test scripts.

Selenium Webdriver is platform-independent since the same code can be used on different Operating Systems like Microsoft Windows, Apple OS and Linux.

When to automate?

- Frequent regression testing, whenever a new feature introduced to application we perform functional testing at the same time we also required to perform regression testing to ensure that new did not affect the existing functionality negatively
- Repeated test case Execution is required, we may need to test few functionalities again and again for set of data like login details
- User Acceptance Tests, we write our code on test environment but we actually have to run the same code on UAT as well to check for bugs quickly
- Faster Feedback to the developers, when developers deployed something tester must be able to give the details of the deployment effect on basic functionalities of the application
- Reduce the Human Effort, when we test a functionality manually it takes lot of time but automating reduces the effort
- Test same application on multiple environments.

What type of tests we can automate with selenium

Selenium supports Functional Testing and Regression Testing of Web Applications

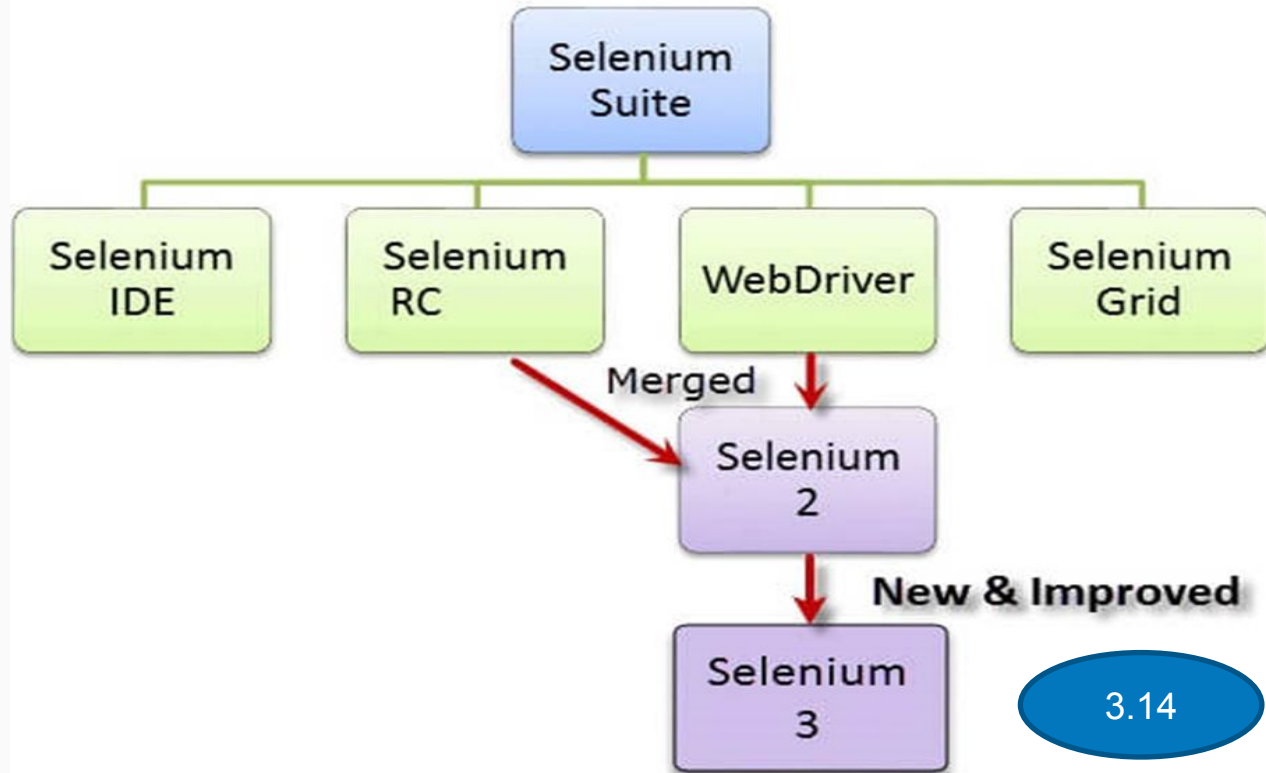
- Functional Testing which includes Smoke Testing and Regression Testing
- Database Testing using drivers like JDBC can also be performed as it is a part of Functional Testing
- Browser Compatibility Testing (Cross browser and Different versions of same browser)
- UI Testing
- Api Testing

Selenium

Selenium has 4 components:

1. Selenium Integrated Development Environment (IDE) [outdated]
1. Selenium Remote Control (RC) [outdated]
1. Selenium WebDriver
1. Selenium Grid

Selenium Component upto selenium 3.X



Selenium IDE

Selenium Integrated Development Environment (IDE) is the simplest framework in the Selenium suite and is the easiest one to learn. It is a Firefox plugin that you can install as easily as you can with other plugins. However, because of its simplicity, Selenium IDE should only be used as a prototyping tool. If you want to create more advanced test cases, you will need to use either Selenium RC or WebDriver.

Selenium IDE

Pros/Cons



PROS

Very easy to use and install.

No programming experience is required, though knowledge of HTML and DOM are needed.

Can export tests to formats usable in Selenium RC and WebDriver.

Has built-in help and test results reporting module.

Provides support for extensions.

CONS

Available only in Firefox.

Designed only to create prototypes of tests.

No support for iteration and conditional operations.

Test execution is slow compared to that of Selenium RC and WebDriver.

Selenium RC

Selenium RC was the flagship testing framework of the whole Selenium project for a long time. This is the first automated web testing tool that allowed users to use a programming language they prefer. As of version 2.25.0, RC can support the following programming languages:

Java

C#

PHP

Python

Perl

Ruby

Selenium RC

Pros/Cons

Pros

- Cross-browser and cross-platform
- Can perform looping and conditional operations
- Can support data-driven testing
- Has matured and complete API
- Can readily support new browsers
- Faster execution than IDE

Cons

- Installation is more complicated than IDE
- Must have programming knowledge
- Needs Selenium RC Server to be running
- API contains redundant and confusing commands
- Browser interaction is less realistic
- Inconsistent results & Uses Javascript
- Slower execution time than WebDriver

Selenium WebDriver

Brief Introduction WebDriver The WebDriver proves itself to be better than both Selenium IDE and Selenium RC in many aspects. It implements a more modern and stable approach in automating the browser's actions. WebDriver, unlike Selenium RC, does not rely on JavaScript for Automation. It controls the browser by directly communicating with it. The supported languages are the same as those in Selenium RC.

Selenium WebDriver Pros/Cons

Pros

- Simpler installation than Selenium RC
- Communicates directly to the browser
- Browser interaction is more realistic
- No need for a separate component such as the RC Server
- Faster execution time than IDE and RC

Cons

- Installation is more complicated than Selenium IDE
- Requires programming knowledge
- Cannot readily support new browsers
- Has no built-in mechanism for logging runtime messages and generating test results

Selenium Grid

Selenium Grid

Selenium Grid is a tool **used together with Selenium RC to run parallel tests** across different machines and different browsers all at the same time. Parallel execution means running multiple tests at once.

Features:

- Enables **simultaneous running of tests in multiple browsers and environments.**
- **Saves time** enormously.
- Utilizes the **hub-and-nodes** concept. The hub acts as a central source of Selenium commands to each node connected to it.

Advantages of Selenium over QTP(UFT)

Quick Test Professional(QTP) is a proprietary automated testing tool previously owned by the company Mercury Interactive before it was acquired by Hewlett-Packard in 2006

Advantages of Selenium over QTP

Selenium	QTP
Open source, free to use, and free of charge.	Commercial.
Highly extensible	Limited add-ons
Can run tests across different browsers	Can only run tests in Firefox, Internet Explorer and Chrome
Supports various operating systems	Can only be used in Windows
Supports mobile devices	QTP Supports Mobile app test automation (iOS & Android) using HP solution called - HP Mobile Center
Can execute tests while the browser is minimized	Needs to have the application under test to be visible on the desktop
Can execute tests in parallel .	Can only execute in parallel but using Quality Center which is again a paid product.

Advantages of QTP over Selenium

Advantages of QTP over Selenium

QTP	Selenium
Can test both web and desktop applications	Can only test web applications
Comes with a built-in object repository	Has no built-in object repository
Automates faster than Selenium because it is a fully featured IDE.	Automates at a slower rate because it does not have a native IDE and only third party IDE can be used for development
Data-driven testing is easier to perform because it has built-in global and local data tables.	Data-driven testing is more cumbersome since you have to rely on the programming language's capabilities for setting values for your test data
Can access controls within the browser (such as the Favorites bar, Address bar, Back and Forward buttons, etc.)	Cannot access elements outside of the web application under test
Provides professional customer support	No official user support is being offered.
Has native capability to export test data into external formats	Has no native capability to export runtime data onto external formats
Parameterization Support is built	Parameterization can be done via programming but is difficult to implement.
Test Reports are generated automatically	No native support to generate test /bug reports.

Advantages Selenium WebDriver

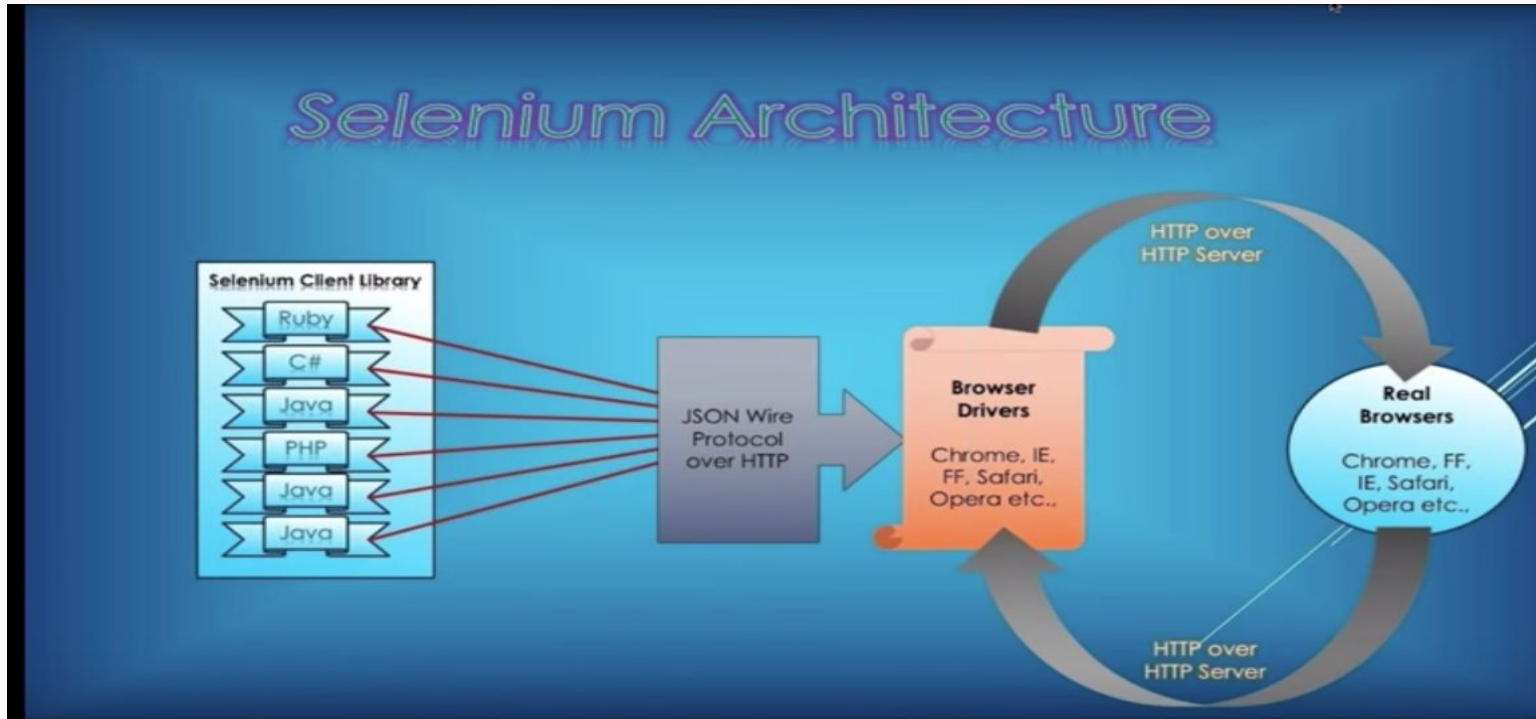
- Selenium webdriver is pure open source, freeware and portable tool
- Selenium webdriver supports many operating systems like Windows, Macintosh, Linux, Unix etc (platform independent)
- Selenium can be integrated with continuous integration tools like Jenkins or Bamboo
- Selenium Webdriver support third party tools like AutoIt, Sikuli, Apache POI, with help of the language
- It supports all popular web browsers
- It supports parallel test execution
- It can be utilized for regression, user interface, functional and UAT testing.
- Supports to take screenshot of execution

Disadvantages Selenium WebDriver

- Since it is open source tool, No reliable Technical support (Official Users Group, Chat room in seleniumhq.org)
- It doesn't support Desktop Applications/Windows bases applications
- No Other tool integration for Test Management
- Image verification is not possible
- Additional Tools Required for Generating Reports.
- New features may not work properly(with new browser driver executable like gecko, chrome functionalities like dragging mouse movement).
- Good tests require good coders.

Selenium Webdriver Architecture

Before starting the automation using any automation tool, it is very important to know how that tool works and how it is architecture.



Selenium webdriver architecture divided into 3 parts

1) **Language Level Bindings :**

Bindings are language level bindings and with which we can implement the Selenium webdriver code.

In simple words these the languages will interact with the Selenium Webdriver and work on various browsers and other devices.

So we have a common API that we use for Selenium that has a common set of commands and we have various bindings for the different languages.

There's Java, Java, Python, Ruby, there's also some other bindings and new bindings can be added very easily.

Selenium webdriver architecture divided into 3 parts

2) **Selenium Webdriver API:**

Now these bindings communicate with Selenium Webdriver API and this API send the commands taken from language level bindings interpret it and sent it to Respective driver.

In basic term it contains set of common library which allow to send command to respective drivers.

3) **Drivers:**

We have various internet browser specific drivers such as IE driver, a Firefox, Chrome, and other Drivers such as HTML.

How all blocks work together?

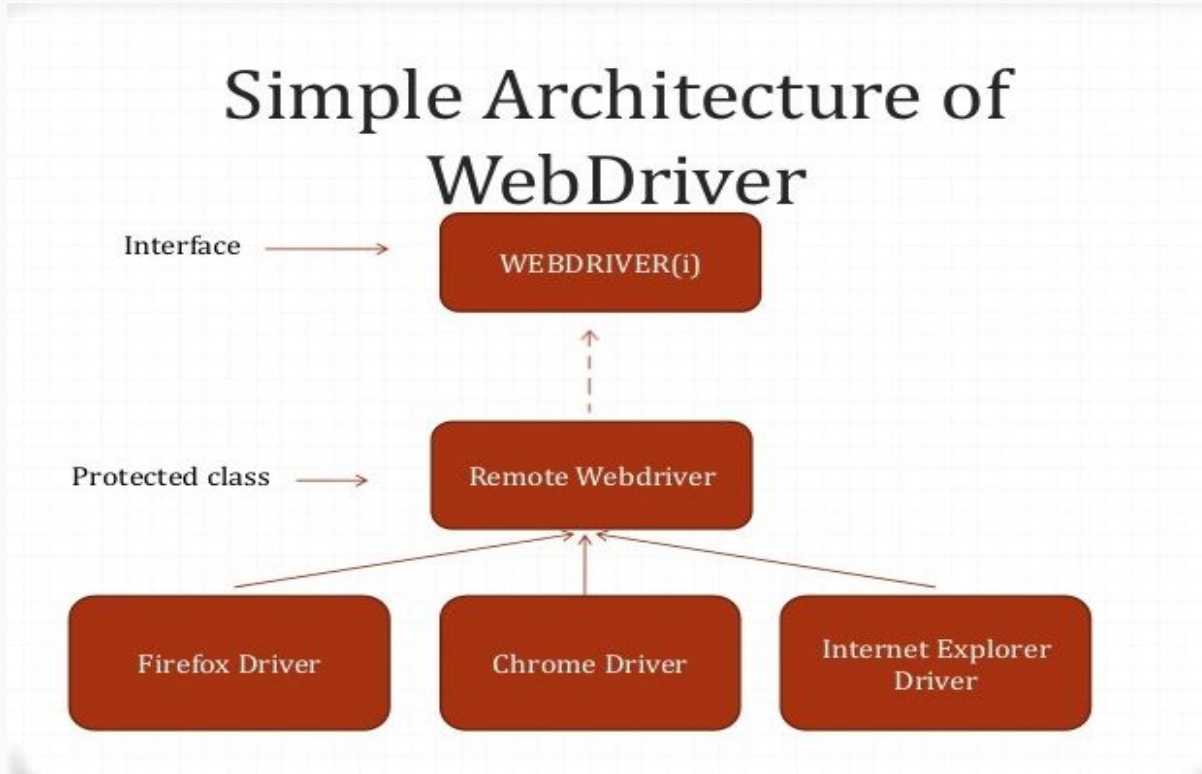
So what's happening here we are going to write test in Java and we are going to be using common **Selenium API** and that **Java binding** is going to be sending command across this common WebDriver API.

Now on the other end is going to be listening a driver, It's going to interpret those commands and it's going to execute them on the actual browser and then it's going to return the result backup using the WebDriver API to our code where we can look at that result.

WebDriver Cannot Readily Support New Browsers

- Remember that WebDriver operates on the OS level. Also, remember that different browsers communicate with the OS in different ways.
- If a new browser comes out, it may have a different process of communicating with the OS as compared to other browsers. So, **we have to give the WebDriver team quite some time to figure that new process out** before they can implement it on the next WebDriver release.
- However, it is up to the WebDriver's team of developers to decide if they should support the new browser or not.

Selenium Webdriver Architecture



First Selenium WebDriver Script

As selenium webdriver supports only web based application, opening a browser for operation is must.

We cannot access already opened browser in selenium

To open Chrome Browser

```
WebDriver driver=new ChromeDriver();
```

To open Firefox Browser

```
WebDriver driver=new FirefoxDriver();
```

To open IE Browser

```
WebDriver driver=new InternetExplorerDriver();
```

First Selenium WebDriver Script

WebDriver - webdriver is the interface which is inherited from SearchContext

new - new is the keyword in java which creates an object (address space) in the heap area of CPU

FirefoxDriver() - **FirefoxDriver()** is a constructor of FirefoxDriver class which implements all the methods in the present in the webdriver interface and this opens the firefox Browser .

driver - driver is reference variable ,which refers the address space created on heap.

Steps

1. Set System Property

```
System.setProperty(key, value)
```

1. Instantiate objects and variables

```
WebDriver driver=new ChromeDriver();
```

1. Launch a Browser Session

WebDriver's `get()` method is used to launch a new browser session and directs it to the URL that you specify as its parameter.

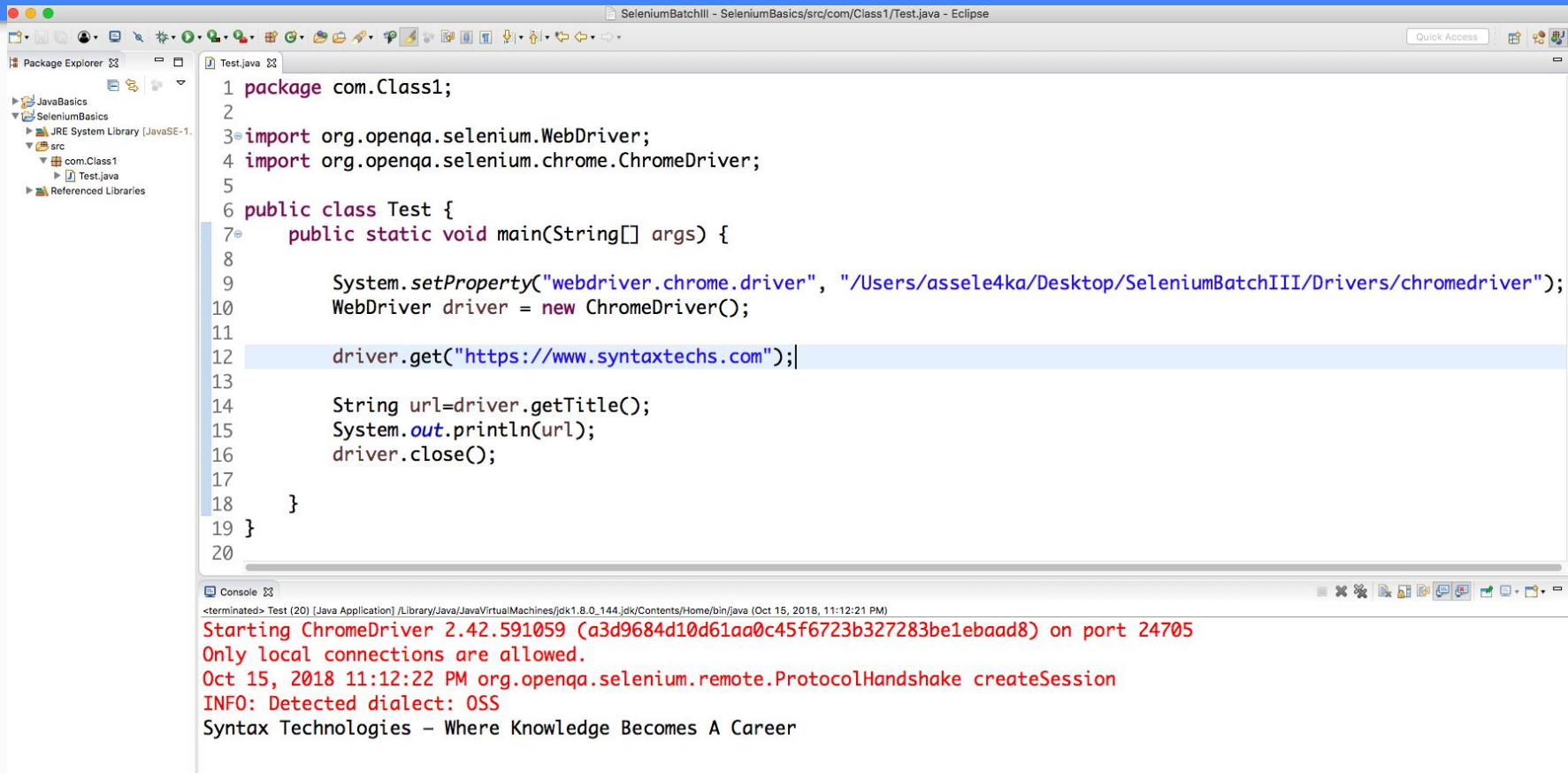
1. Get the Actual Page Title

The `WebDriver` class has the `getTitle()` method that is always used to obtain the page title of the currently loaded page.

1. Terminate a Browser Session

The `"close()"` method is used to close the browser window.

Chrome Demo



```
1 package com.Class1;
2
3 import org.openqa.selenium.WebDriver;
4 import org.openqa.selenium.chrome.ChromeDriver;
5
6 public class Test {
7     public static void main(String[] args) {
8
9         System.setProperty("webdriver.chrome.driver", "/Users/assele4ka/Desktop/SeleniumBatchIII/Drivers/chromedriver");
10        WebDriver driver = new ChromeDriver();
11
12        driver.get("https://www.syntaxtechs.com");|
13
14        String url=driver.getTitle();
15        System.out.println(url);
16        driver.close();
17
18    }
19 }
20
```

Console

```
<terminated> Test (20) [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_144.jdk/Contents/Home/bin/java (Oct 15, 2018, 11:12:21 PM)
Starting ChromeDriver 2.42.591059 (a3d9684d10d61aa0c45f6723b327283be1ebaad8) on port 24705
Only local connections are allowed.
Oct 15, 2018 11:12:22 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: OSS
Syntax Technologies - Where Knowledge Becomes A Career
```

Firefox Demo

The screenshot displays an IDE window with a Java file named `Test.java`. The code is as follows:

```
1 package com.Class1;
2
3 import org.openqa.selenium.WebDriver;
4 import org.openqa.selenium.firefox.FirefoxDriver;
5
6 public class Test {
7     public static void main(String[] args) {
8
9         System.setProperty("webdriver.gecko.driver", "/Users/assele4ka/Desktop/SeleniumBatchIII/Drivers/geckodriver");
10        WebDriver driver = new FirefoxDriver();
11
12        driver.get("https://www.syntaxtechs.com");
13
14        String url=driver.getTitle();
15        System.out.println(url);
16        driver.close();
17
18    }
19 }
20
```

The Package Explorer on the left shows the project structure:

- JavaBasics
 - SeleniumBasics
 - JRE System Library [JavaSE-1.8]
 - src
 - com.Class1
 - Test.java
- Referenced Libraries

The Console window at the bottom shows the following output:

```
<terminated> Test [20] [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_144.jdk/Contents/Home/bin/java (Oct 15, 2018, 11:25:47 PM)
1539660350620 Marionette DEBUG [4294967297] Received DOM event beforeunload for about:blank
2018-10-15 23:25:50.825 plugin-container[49774:10949178] *** CFMessagePort: bootstrap_register(): failed 1100 (0x44c) 'Perr
See /usr/include/servers/bootstrap_defs.h for the error codes.
1539660351162 Marionette DEBUG [4294967297] Received DOM event pagehide for about:blank
1539660355937 Marionette DEBUG [4294967297] Received DOM event DOMContentLoaded for https://www.syntaxtechs.com/
1539660356592 Marionette DEBUG [4294967297] Received DOM event pageshow for https://www.syntaxtechs.com/
Syntax Technologies - Where Knowledge Becomes A Career
1539660356602 Marionette INFO Stopped listening on port 51644
```