**Selenium Syllabus**

**Selenium Introduction**

**Installation of Java, Eclipse and Selenium jars download**

**Configuration of Selenium Jars into Eclipse.**

**Webdriver Program on Different Webdriver methods.**

**How to Invoke different browsers using Webdriver.**

**Different Locator Techniques and tools used.**

* **Firebug and Firepath Add-ons**
* **Element Identification by different locating mechanism.**
* **Advanced ways of locating webelement/objects**
* **Writing customized xpath and cssselector.**

**Techniques to automate web UI**

* **Handling Dropdown/Select Statement**
* **Handling checkboxes, radio buttons.**
* **Handling Javascript alert.**

**Web Element Validation techniques.**

* **Techniques to automate WebUI**
* **Handling Ajax/Mouse interaction**
* **Handling window, multiple windows, Frames**

**Cross Browser Testing Selenium Grid**

**TestNG**

**Java OOPS Basics for Selenium**

**Selenium Introduction**

**Selenium** is an open source automation testing tool.  
It is used exclusively for web-based applications.

You can work on multiple operating systems using selenium.

Platforms Supported by Selenium

* Windows
* OS X
* Linux
* Solaris

following languages are used with selenium.

- Java  
- C#  
- Ruby  
- Python  
- PHP  
- Perl

Selenium Browsers Support:

* Internet Explorer
* Firefox
* Chrome
* Safari

**Selenium Features**

What makes Selenium Unique from other Automation tools?

Firstly, selenium is an open source technology, which makes it more preferable than QTP (Quick Test Professional) which is a licensed one.

Here are the few reasons why selenium is more popular than other testing tools

* Selenium supports Java, Python, Pearl, Ruby, PHP, .NET (C#) and many other programming languages. You required to be comfortable with any one language in order to operate selenium
* You can develop your test cases using IDE’S that you are already familiar with like Eclipse, Visual Studio, NetBeans.
* Selenium has capability to operate on almost every OS.
* Selenium has suite of tools among which Selenium-Grid allows you to run your tests on different machines against different browsers in parallel. That is, running multiple tests at the same time against different machines running different browsers and operating systems.
* Essentially, Selenium-Grid support distributed test execution. It allows for running your tests in a distributed test execution environment. This reduces the time it takes for the test suite to complete a test pass.
* One highly beneficial feature of Selenium is that the language used for building the program is independent of the language that the web application or website is using. This implies that the test script can be developed in any of the languages that Selenium supports.
* This testing tool supports a range of browsers like Opera, Safari, Chrome, IE 6, 7, 8 and Mozilla Firefox.
* Selenium has a very dynamic developer community that is backed by Google.
* So far as Selenium is concerned, we have a number of robust methods for location of elements such as CSS, Xpath, DOM, and so on.
* With Selenium, it is convenient to implement frameworks that revolve around Object oriented programming like Keyword Driven, Data driven and Hybrid.
* Selenium provides support for integration of open source frameworks like TestNG, JUnit, NUnit and so on.
* With the use of Selenium, it is possible to execute simultaneous tests leveraging various browsers on various machines. This is turn cuts down the time for test execution when a large project is in progress.

Explain WebDriver Architecture.

* Architecture of Selenium WebDriver is all about how Selenium works internally. We know Selenium is a browser automation tool which interacts with browser and automate end to end tests of a web application.
* Selenium is a suite of tools. It consists of Selenium IDE, Selenium RC, Selenium Webdriver, and Selenium Grid.

**Selenium IDE:**

Selenium IDE (Integrated Development Environment) is a Firefox plugin. It is the simplest framework in the Selenium Suite. It allows us to record and playback the scripts. Even though we can create scripts using Selenium IDE, we need to use Selenium RC or [Selenium WebDriver](https://www.softwaretestingmaterial.com/selenium-tutorial/) to write more advanced and robust [test cases](https://www.softwaretestingmaterial.com/test-case-template-with-explanation/).

**Drawbacks:**

* As it will come with only Firefox addin. What if you want to test your application which works only in Internet explorer or some any other browser?
* Selenium IDE is not suitable when you want to built a robust frameworks.
* Selenium IDE doesn’t provide iteration or conditional statements for test scripts.

**Selenium RC:**

Selenium RC AKA Selenium 1. Selenium RC was the main Selenium project for a long time before the WebDriver merge brought up Selenium 2. Selenium 1 is still actively supported (in maintenance mode). It relies on JavaScript for automation. It supports Java, JavaScript, Ruby, PHP, Python, Perl and C#. It supports almost every browser out there.

**Drawbacks.**

* As it is entirely use JavaScript to talk to browser, it leads to significant weakness. Every browser impose very strict security rules on the JavaScript being executed to protect the users from malicious scripts
* There is no support for Android and IOS Platform
* Server need to be started every time to run a program.
* Lot of Limitations when an Application has Rich API with dynamic elements
* Native keyboard and mouse events cannot be handled in efficient manner

**Note:** Selenium RC is officially deprecated.

**Selenium WebDriver(2.0):**

Selenium WebDriver is a browser automation framework that accepts commands and sends them to a browser. It is implemented through a browser-specific driver. It controls the browser by directly communicating with it. Selenium WebDriver supports Java, C#, PHP, Python, Perl, Ruby.

**Operation System Support** – Windows, Mac OS, Linux, Solaris  
**Browser Support** – Mozilla Firefox, Internet Explorer, Google Chrome 12.0.712.0 and above, Safari, Opera 11.5 and above, Android, iOS, HTML Unit 2.9 and above.

**Advantages:**

* In addition, Selenium 2 still runs Selenium 1’s Selenium RC interface for backwards compatibility.
* No server required to start.
* No support for Android and iPhone platform in RC
* Can Handle rich API
* Can handle Mouse movements
* It directly talks to browser. unlike RC it does not use any proxy server
* It supports all the Latest Versions of Firefox
* All future enhancements can be done in Webdriver only.

**Selenium Grid:**

Selenium Grid is a tool used together with Selenium RC to run tests on different machines against different browsers in parallel. That is, running multiple tests at the same time against different machines running different browsers and operating systems.

Selenium WebDriver is a well-designed object-oriented API supports many languages such as Java, C#, Python etc. Let’s see what is API and then we move on to know the complete picture of Selenium WebDriver Architecture.

**API:**

Application Programming Interface (API) works as an interface between various software components.

**Selenium WebDriver API:**

Selenium Webdriver API helps in communication between languages and browsers. Selenium supports many programming languages such as Java, C#, Python etc., also it supports multiple browsers such as Google Chrome, Firefox, Internet Explorer etc.,

**Installation of Java, Eclipse and Selenium jars download**

* Download Java from below link(<http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html>)
* Set Java Home Path in System variables for Windows/Mac OS
* Install Eclipse and do the configuration.
* Create new Eclipse project. (First Project is Test Suite, Selenium API in Java language, Java class file – Testcase)
* Download the selenium jar files from the below link.[(http://www.seleniumhq.org)]((http:/www.seleniumhq.org)).
* Configure Selenium jars in the Project Build path.
* Choose the browser to run.
* Create Driver object based on browser chosen
* Set System property of the Browser
* Run your first Program.

**Webdriver Program on Different Webdriver methods.**

I**nterface**:

A Java interface defines a set of methods but does not implement them. A class that implements the interface agrees to implement all of the methods defined in the interface.

**Webdriver is the Interface**

**All Known Implementing Classes:**

[AndroidDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/android/AndroidDriver.html), [AndroidWebDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/android/library/AndroidWebDriver.html), [ChromeDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/chrome/ChromeDriver.html), [EventFiringWebDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/support/events/EventFiringWebDriver.html), [FirefoxDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/firefox/FirefoxDriver.html), [HtmlUnitDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/htmlunit/HtmlUnitDriver.html), [InternetExplorerDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/ie/InternetExplorerDriver.html),[IPhoneDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/iphone/IPhoneDriver.html), [IPhoneSimulatorDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/iphone/IPhoneSimulatorDriver.html), [RemoteWebDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/remote/RemoteWebDriver.html), [SafariDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/safari/SafariDriver.html)

**Steps for Creating class object with reference to interface**

1. Choose the Browse on which you want to perform Automation and Select the respective class to implement Webdriver

**Ex:** [FirefoxDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/firefox/FirefoxDriver.html)()

**2.** Create an object for the Class and assign name to it

**Selenium=New** [FirefoxDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/firefox/FirefoxDriver.html)();

3.Then make Class object reference to the Webdriver Interface which you want to implement

Webdriver **Selenium=New** [FirefoxDriver](http://selenium.googlecode.com/svn/trunk/docs/api/java/org/openqa/selenium/firefox/FirefoxDriver.html)();

**Basis methods of Webdriver**:

**Get()**;

Load a new web page in the current browser window.

**getCurentURl();**  
Get a string representing the current URL that the browser is looking at.

**getTitle()**

The title of the current page

**getPageSource()**

Get the source of the last loaded page.

**Quit();**

Quits this driver, closing every associated window.

**Close();**

Close the current window, quitting the browser if it's the last window currently open.

Geckodriver for firefox...

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

When you are using Selenium 3 releases, you have to download geckodriver to run your test on firefox browser. Just like the other drivers(chromedriver/iedriver) available to Selenium, Mozilla has released geckodriver executable that will run alongside the firefox browser. Firefox version 48 & above will use geckodriver.

If you want to work with Firefox you have to set the property now. You can download geckodriver from Github and then you can extract and you will get .exe file. https://github.com/mozilla/geckodriver/releases/

You can place it on C:\\ and use as below -

Sample Code:-

System.setProperty("webdriver.gecko.driver”,”C:\\geckodriver.exe");

WebDriver driver = new FirefoxDriver();

driver.get("http://www.google.com");

driver.quit();