**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  
- Pyton  
- PHP  
- Perl

**Selenium Browsers Support:**

InternetExplorer

Firefox

Chrome

Safari

**SeleniumHistory**

Selenium came into existence in 2004 when a GUY named Jason Huggins was testing an internal application at ThoughtWorks.

Selenium is a set of different software tools each with a different approach to supporting test automation.

**Selenium Components:**

[Selenium IDE](http://en.wikipedia.org/wiki/Selenium_(software)#Selenium_IDE)

[Selenium Remote Control](http://en.wikipedia.org/wiki/Selenium_(software)#Selenium_Remote_Control)

[Selenium WebDriver](http://en.wikipedia.org/wiki/Selenium_(software)#Selenium_WebDriver)

[Selenium Grid](http://en.wikipedia.org/wiki/Selenium_(software)#Selenium_Grid)

Let’s have a look on each of these components:

[**Selenium IDE**](http://en.wikipedia.org/wiki/Selenium_(software)#Selenium_IDE)

Selenium IDE is a complete [integrated development environment](http://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) for Selenium tests. It is implemented as a [Firefox extension](http://en.wikipedia.org/wiki/Add-on_(Mozilla)), and allows recording, editing, and debugging tests. It was previously known as Selenium Recorder.

Scripts are recorded in Selenese, a special test scripting language for Selenium. Selenese provides commands for performing actions in a browser (click a link, select an option), and for retrieving data from the resulting pages.

It is not only just play back tool, It can records user actions as they are performed and then exports them as reusable script in one of many programming languages that can be later executed.

**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 1.0)**

Selenium RC was first component in the market which control a browser from a language of your choice. The underlying functionality of Selenium Remote Control is that it uses Javascript library (Selenium core) that could drive interactions with the page, allows us to automatically rerun tests against multiple browsers.

**Selenium RC Architecture**

There is a selenium-RC server which acts as proxy between our Driver code and Application under Test

The client/driver establishes a connection with the selenium-RC server.  
2. Selenium RC server launches a browser (or reuses an old one) with a URL that injects Selenium-Core’s JavaScript into the browser-loaded web page.  
3. The client-driver passes a Selenese command to the server.  
4. The Server interprets the command and then triggers the corresponding JavaScript execution to execute that command within the browser.  
5. Selenium-Core instructs the browser to act on that first instruction, typically opening a page of the AUT.  
6. The browser receives the open request and asks for the website’s content from the Selenium RC server (set as the HTTP proxy for the browser to use).  
7. Selenium RC server communicates with the Web server asking for the page and once it receives it, it sends the page to the browser masking the origin to look like the page comes from the same server as Selenium-Core (this allows Selenium-Core to comply with the Same Origin Policy).  
8. The browser receives the web page and renders it in the frame/window reserved for it

**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 a Application has Rich API with dynamic elements

Native keyboard and mouse events cannot be handled in efficient manner

**Webdriver: (Selenium 2.0)**

This brand new automation tool provides all sorts of features, including a more cohesive and object oriented API as well as an answer to the limitations of the old implementation. It supports the WebDriver API underlying technology, along with the Selenium underlying technology,

The architecture of Selenium Webdriver is entirely different from RC. Unlike RC there is no proxy server between AUT and Code.

It makes direct calls to browser native API to get the things executed. Unlike RC it does not Use any proxy server to talk to browser. WebDriver uses the most appropriate way to access the browser API. If we look at Firefox, it uses JavaScript to access the API. If we look at Internet Explorer, it uses C++. That means that it sometimes needs direct help from browser development team. By this approach we can control browsers in the best possible way but has the downside that new browsers entering the market will not be supported straight away like we can with RC.

As Webdriver directly talks with browser we can overcome the limitations of JavaScript security model which we have faced with Selenium Core in RC

Selenium-WebDriver supports the following browsers along with the operating systems these browsers are compatible with.

• Google Chrome 12.0.712.0+

• Internet Explorer 6, 7, 8, 9 - 32 and 64-bit where applicable

• Firefox 3.0, 3.5, 3.6, 4.0, 5.0, 6, 7

• Opera 11.5+

• HtmlUnit 2.9

• Android – 2.3+ for phones and tablets (devices & emulators)

• iOS 3+ for phones (devices & emulators) and 3.2+ for tablets (devices & emulators)

Languages on which Webdriver Supports:

Java,

Javascript,

Ruby,

PHP,

Python,

Perl

C#

You can use any of the above language to Automate application. There is nothing rule like as your website is built with C#,so you have to use only C# code in Webdriver to automate your application.The language you write your code is Application independent.

**Why Webdriver??**

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.

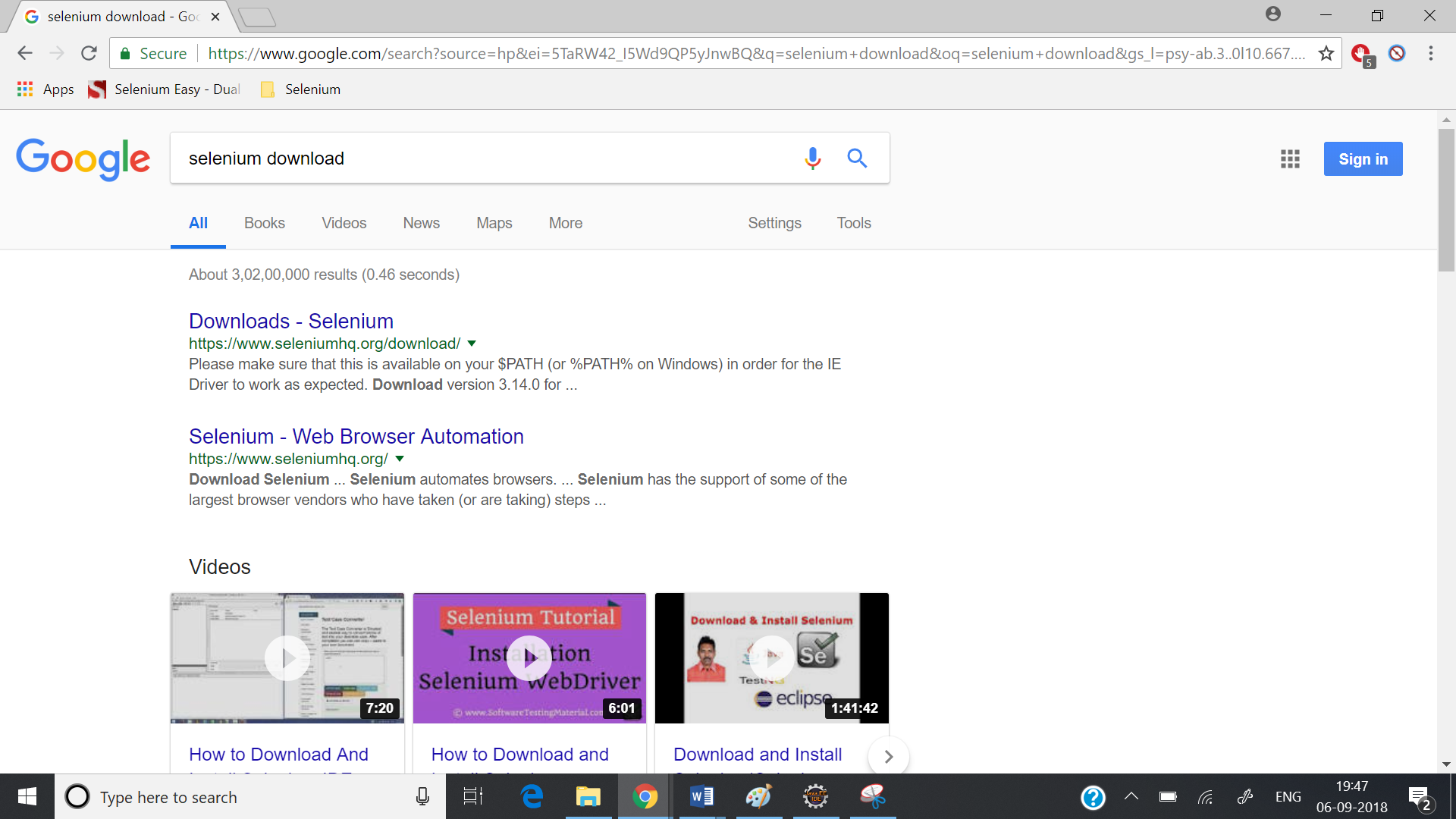
[**Selenium Grid**](http://en.wikipedia.org/wiki/Selenium_(software)#Selenium_Grid)

Selenium Grid allows you to run your tests in parallel, that is, different tests can be run at the same time on different remote machines. Also, if you must run your test suite on multiple environments you can have different remote machines supporting and running your tests in them at the same time. In each case Selenium Grid greatly improves the time it takes to run your suite by making use of parallel processing

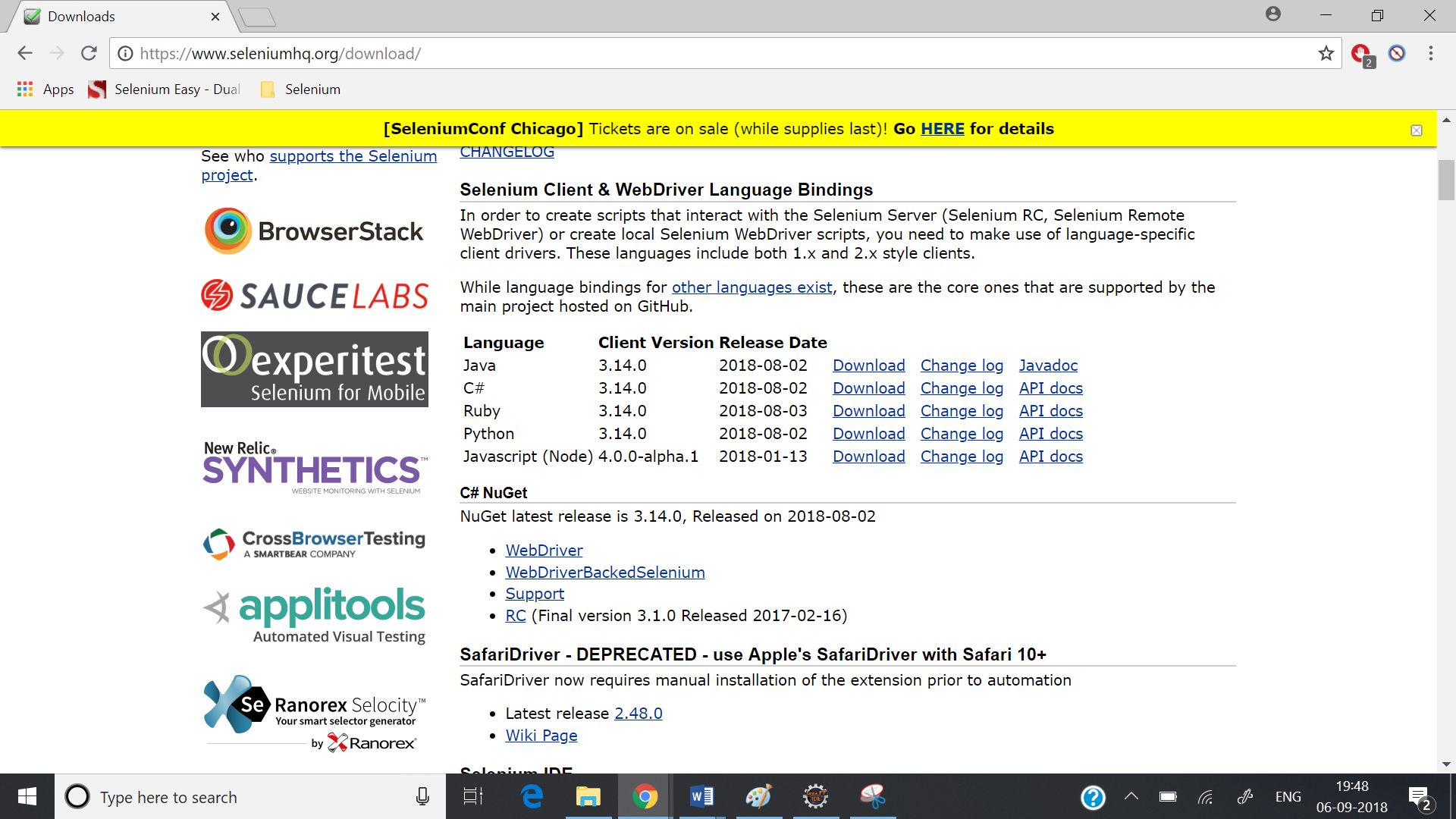
**Downloading and Configuring Selenium:**

Once we write selenium code, first the program should identify that it is selenium code. In order for the program to identify it as selenium code, first we need to download selenium and create a link between selenium and project we have created.

Go to google and type “Selenium Download”



Go to the link “Downloads – Selenium”

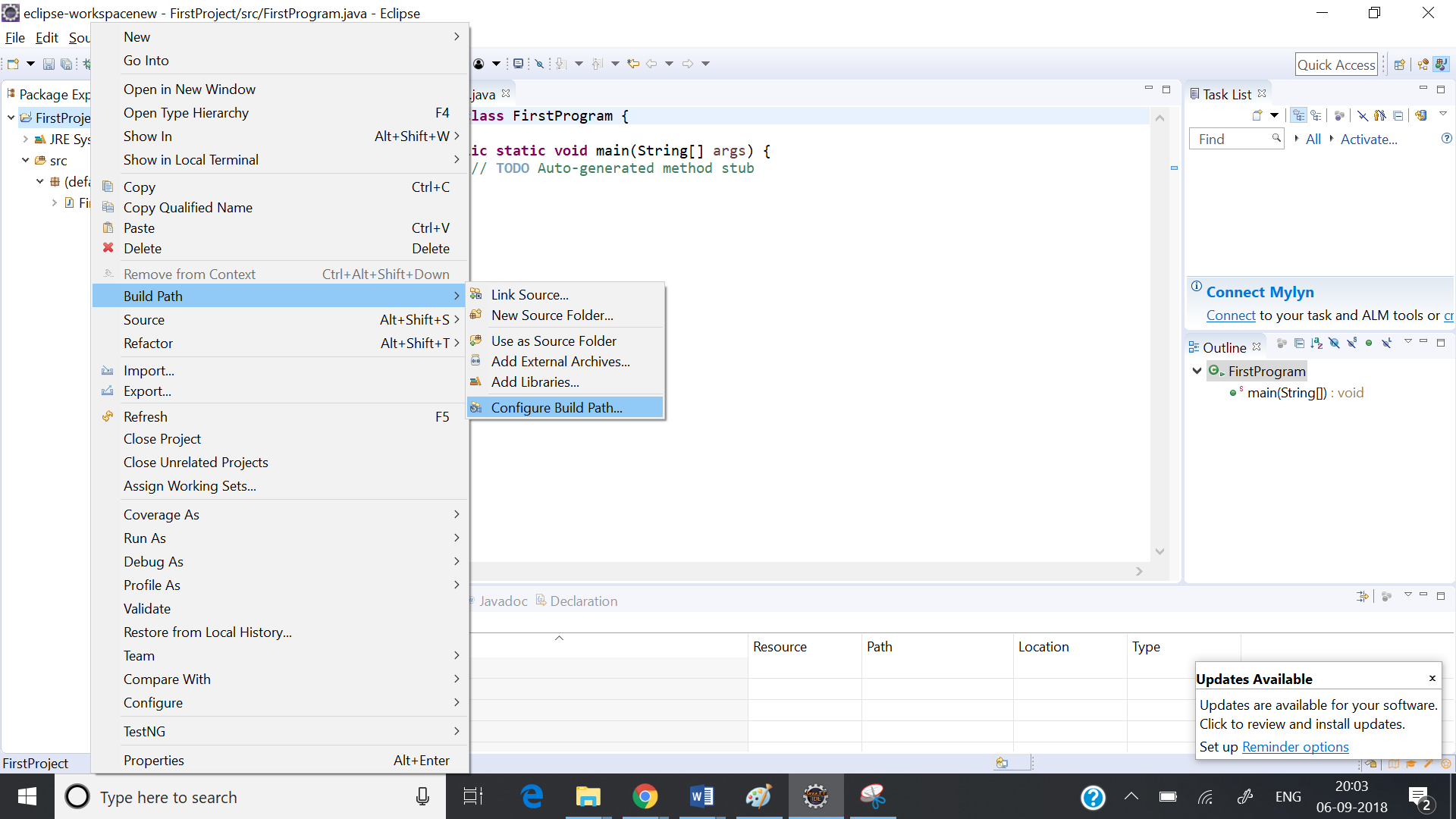


Download Java related jar files.

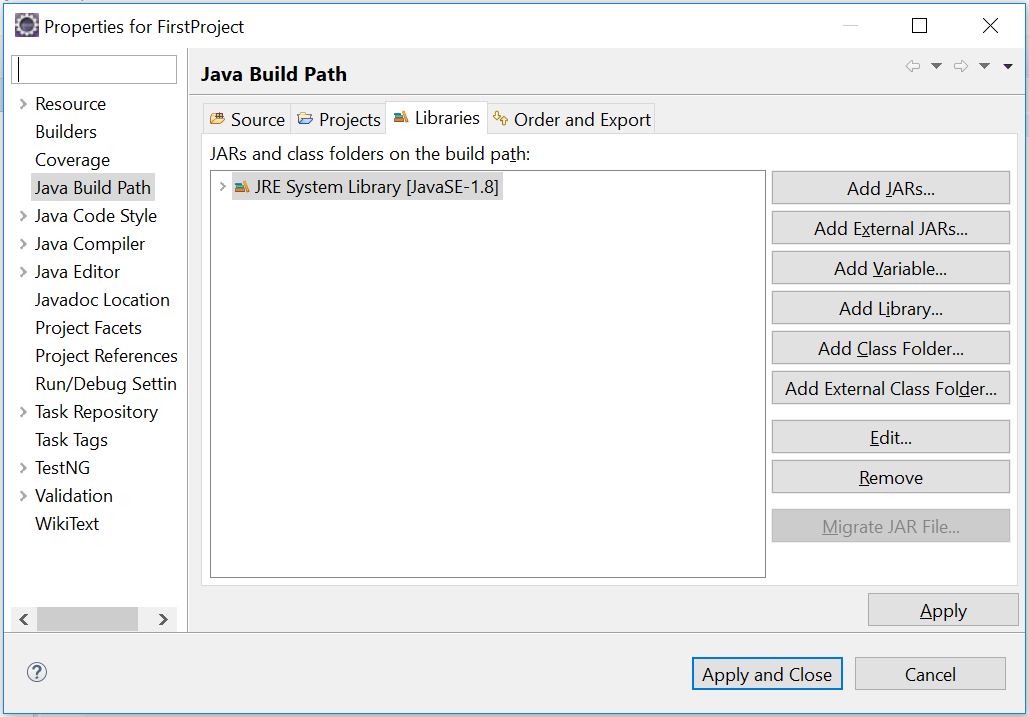
Once the files are downloaded, unzip them and you will see some jar files.

All these jar files need to be linked to your project.

**Linking the jar files to your selenium project:**



Right click on the project go to Build Path and go to Configure build. A new window opens and then go to Java Build Path and click on the tab “Libraries”.



Click on Add External Jars and add all the selenium jar files we have downloaded. Then click on “Apply and Close”.