**What is Selenium Webdriver?**

**Selenium Webdriver** is an open-source collection of APIs which is used for testing web applications. The Selenium Webdriver tool is used for automating web application testing to verify that it works as expected or not. It mainly supports browsers like Firefox, Chrome, Safari and Internet Explorer. It also permits you to execute cross-browser testing.

WebDriver also enables you to **use a programming language** in creating your test scripts (not possible in Selenium IDE).

You can now use **conditional operations** like if-then-else or switch-case. You can also perform looping like do-while.



How to write a TEST CASE Software Testing Tutorial

Following programming languages are supported by WebDriver

* [Java](https://www.guru99.com/java-tutorial.html)
* .Net
* [PHP](https://www.guru99.com/php-tutorials.html)
* [Python](https://www.guru99.com/python-tutorials.html)
* [Perl](https://www.guru99.com/perl-tutorials.html)
* Ruby

**You do not have to know all of them. You just need to be knowledgeable in one.**However, in this tutorial, we will be using Java with Eclipse as our IDE.

**Difference between Selenium RC and Webdriver**

Before the advent of WebDriver in  2006, there was another, **automation tool called Selenium Remote Control.**Both WebDriver and Selenium RC have following features:

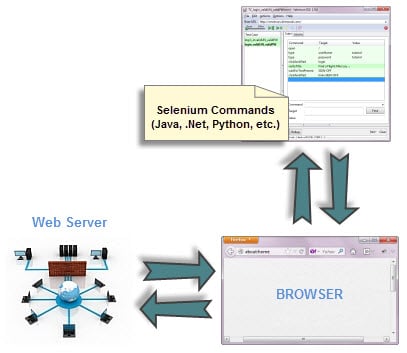
* They both allow you to **use a programming language** in designing your test scripts.
* They both allow you to **run your tests against different browsers.**

So how do they differ? Let us discuss the answers.

**1. Architecture**

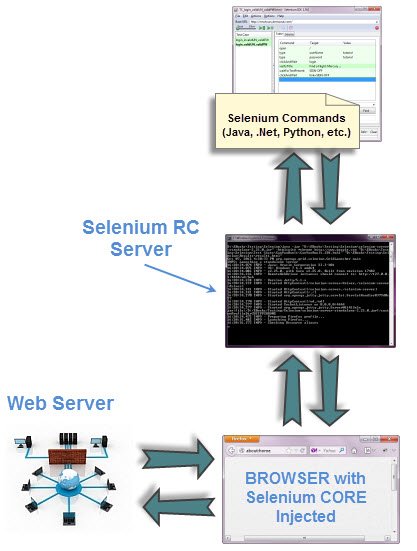
**WebDriver’s architecture is simpler than Selenium RC’s**.

* It controls the browser from the OS level
* All you need are your programming language’s IDE (which contains your Selenium commands) and a browser.



**Selenium RC’s architecture is way more complicated.**

* You first need to launch **a separate application called Selenium Remote Control (RC) Server** before you can start testing
* The Selenium RC Server **acts as a “middleman” between your Selenium commands and your browser**
* When you begin testing, Selenium RC Server “injects” a **Javascript program called Selenium Core** into the browser.
* Once injected, Selenium Core will start receiving instructions relayed by the RC Server from your test program.
* When the instructions are received, **Selenium Core will execute them as Javascript commands.**
* The browser will obey the instructions of Selenium Core and will relay its response to the RC Server.
* The RC Server will receive the response of the browser and then display the results to you.
* RC Server will fetch the next instruction from your test script to repeat the whole cycle.



**2. Speed**



**WebDriver is faster than Selenium RC since it**speaks directly to the browser uses the browser’s own engine to control it.



**Selenium RC is slower since it uses a Javascript program called Selenium Core.**This Selenium Core is the one that directly controls the browser, not you.

**3. Real-life Interaction**

****

**WebDriver interacts with page elements in a more realistic way.** For example, if you have a disabled text box on a page you were testing, WebDriver really cannot enter any value in it just as how a real person cannot.



Selenium Core, just like other[JavaScript](https://www.guru99.com/interactive-javascript-tutorials.html)codes, can access disabled elements. In the past, Selenium testers complain that Selenium Core was able to enter values to a disabled text box in their tests. Differences in API

**4. API**



**Selenium RC’s API is more matured but contains redundancies and often confusing commands**. For example, most of the time, testers are confused whether to use type or typeKeys; or whether to use click, mouseDown, or mouseDownAt. Worse, **different browsers interpret each of these commands in different ways too!**

**WebDriver’s API is simpler than Selenium RC’s**. It does not contain redundant and confusing commands.

**5. Browser Support**



**WebDriver can support the headless HtmlUnit browser**

HtmlUnit is termed as “headless” because it is an invisible browser – it is GUI-less.

It is a very fast browser because no time is spent in waiting for page elements to load. This accelerates your test execution cycles.

Since it is invisible to the user, it can only be controlled through automated means.

**Selenium RC cannot support the headless HtmlUnit browser.**It needs a real, visible browser to operate on.

**Limitations of WebDriver**

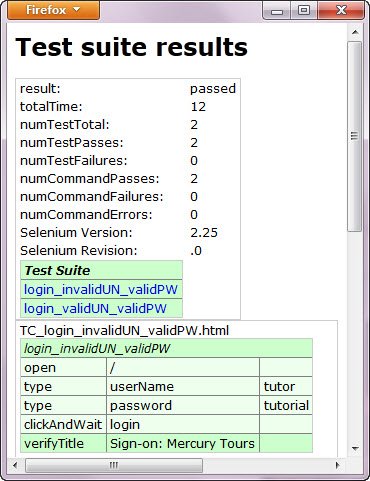
**WebDriver Cannot Readily Support New Browsers**

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, **you 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 RC Has Built-In Test Result Generator**

**Selenium RC automatically generates an HTML file of test results**. The format of the report was pre-set by RC itself. Take a look at an example of this report below.



**WebDriver has no built-in command that automatically generates a Test Results File**. You would have to rely on your IDE’s output window, or design the report yourself using the capabilities of your programming language and store it as text, HTML, etc.