

# Learn Selenium in 1 Day

By Krishna Rungta

Copyright 2018 - All Rights Reserved – Krishna Rungta

**ALL RIGHTS RESERVED.** No part of this publication may be reproduced or transmitted in any form whatsoever, electronic, or mechanical, including photocopying, recording, or by any informational storage or retrieval system without express written, dated and signed permission from the author.

# **Table Of Content**

**Chapter 1: Introduction to Selenium**

**Chapter 2: Install Selenium IDE and FireBug**

**Chapter 3: Introduction to Selenium IDE**

**Chapter 4: Creating your First Selenium IDE script**

**Chapter 5: How to use Locators in Selenium IDE**

**Chapter 6: How to enhance a script using Selenium IDE**

**Chapter 7: Store Variables, Echo, Alert, PopUp handling in Selenium IDE**

**Chapter 8: Introduction to WebDriver & Comparison with Selenium RC**

**Chapter 9: Guide to install Selenium WebDriver**

**Chapter 10: Creating your First Script in Webdriver**

**Chapter 11: Accessing Forms in Webdriver**

**Chapter 12: How to Select Option from DropDown using Selenium Webdriver**

**Chapter 13: Accessing Links & Tables using Selenium Webdriver**

**Chapter 14: Keyboard & Mouse Event using Action Class in Selenium Webdriver**

**Chapter 15: How to Upload & Download a File using Selenium Webdriver**

**Chapter 16: XPath in Selenium: Complete Guide**

**Chapter 17: How TestNG makes Selenium tests easier**

**Chapter 18: Handling Date Time Picker using Selenium**

**Chapter 19: Alert & Popup handling in Selenium**

**Chapter 20: Handling Dynamic Web Tables Using Selenium WebDriver**

**Chapter 21: Using Contains, Sibling, Ancestor to Find Element in Selenium**

**Chapter 22: Implicit & Explicit Waits in Selenium**

**Chapter 23: Parameterization using XML and DataProviders: Selenium**

**Chapter 24: All About Excel in Selenium: POI & JXL**

**Chapter 25: Page Object Model (POM) & Page Factory in Selenium: Ultimate Guide**

**Chapter 26: Introduction to Selenium Grid**

**Chapter 27: Maven & Jenkins with Selenium: Complete Tutorial**

**Chapter 28: Creating Keyword & Hybrid Frameworks with Selenium**

**Chapter 29: Database Testing using Selenium: Step by Step Guide**

**Chapter 30: Handling Iframes in Selenium**

**Chapter 31: Cross Browser Testing using Selenium**

**Chapter 32: PDF , Emails and Screenshot of Test Reports in Selenium**

**Chapter 33: How to Take Screenshot in Selenium WebDriver**

**Chapter 34: Sessions, Parallel run and Dependency in Selenium**

**Chapter 35: Tutorial on Log4j and LogExpert with Selenium**

**Chapter 36: Selenium with HTMLUnit Driver & PhantomJS**

**Chapter 37: Using Robot API with Selenium**

**Chapter 38: How to use AutoIT with Selenium**

**Chapter 39: Desired Capabilities in Selenium**

**Chapter 40: SSL Certificate Error Handling in Selenium**

**Chapter 41: Handling Ajax call in Selenium Webdriver**

**Chapter 42: Listeners and their use in Selenium WebDriver**

**Chapter 43: Execute JavaScript based code using Selenium Webdriver**

**Chapter 44: Using Selenium with Python**

**Chapter 45: How to use IntelliJ & Selenium Webdriver**

**Chapter 46: Test Case Priority in TestNG**

**Chapter 47: TestNG: Execute multiple test suites**

**Chapter 48: Introduction to TestNG Groups**

**Chapter 49: Verify Tooltip Using Selenium WebDriver**

**Chapter 50: Flash Testing with Selenium**

**Chapter 51: How to Find Broken links using Selenium Webdriver**

**Chapter 52: Selenium Core Extensions**

**Chapter 53: Using Apache Ant with Selenium**

**Chapter 54: Using Selenium with Github**

**Chapter 55: Handling Cookies in Selenium WebDriver**

**Chapter 56: Using SoapUI with Selenium**

**Chapter 57: XSLT Report in Selenium**

**Chapter 58: Firefox Profile - Selenium WebDriver**

**Chapter 59: Breakpoints and Startpoints in Selenium**

**Chapter 60: Top 100 Selenium Interview Questions & Answers**

**Chapter 61: Using Cucumber with Selenium**

**Chapter 62: Drag and Drop action in Selenium**

**Chapter 63: Selenium C# Webdriver Tutorial for Beginners**

**Chapter 64: Creating Object Repository in Selenium WebDriver**

**Chapter 65: Scroll UP or Down a page in Selenium Webdriver**

**Chapter 66: File Upload using Sikuli in Selenium Webdriver**

**Chapter 67: Gecko (Marionette) Driver Selenium: Download, Install, Use with Firefox**

**Chapter 68: Find Element and Find Elements in Selenium**

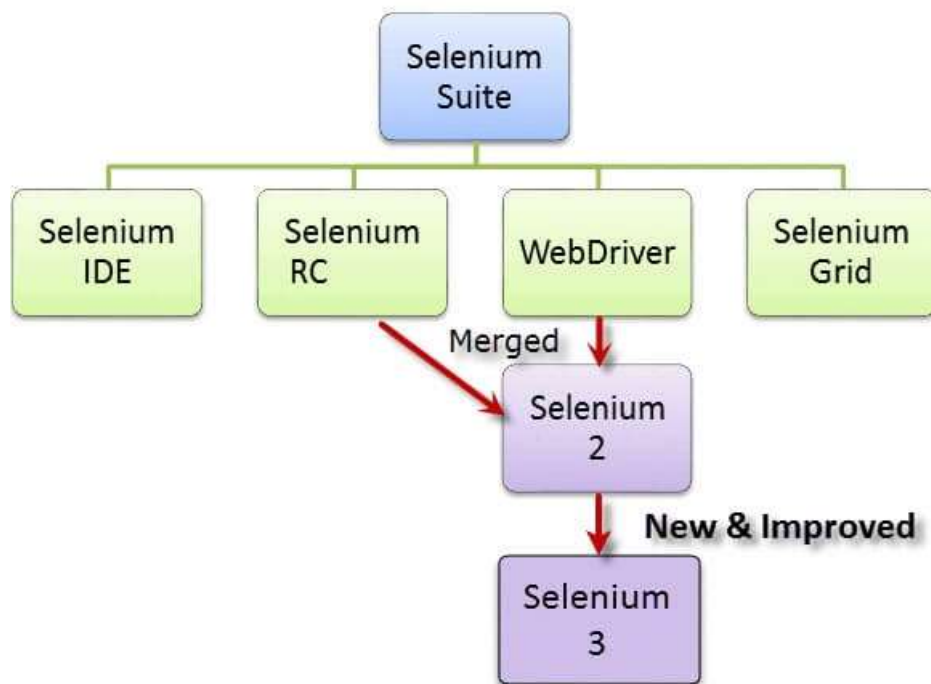
# Chapter 1: Introduction to Selenium

## What is Selenium?

Selenium is a free (open source) automated testing suite for web applications across different browsers and platforms. It is quite similar to HP Quick Test Pro (QTP now UFT) only that Selenium focuses on automating web-based applications. Testing done using Selenium tool is usually referred as Selenium Testing.

Selenium is not just a single tool but a suite of software's, each catering to different testing needs of an organization. **It has four components.**

- Selenium Integrated Development Environment (IDE)
- Selenium Remote Control (RC)
- WebDriver
- Selenium Grid



At the moment, Selenium RC and WebDriver are merged into a single framework to form **Selenium 2**. Selenium 1, by the way, refers to Selenium RC.

## Who developed Selenium?

Since Selenium is a collection of different tools, it had different developers as well. Below are the key persons who made notable contributions to the Selenium Project

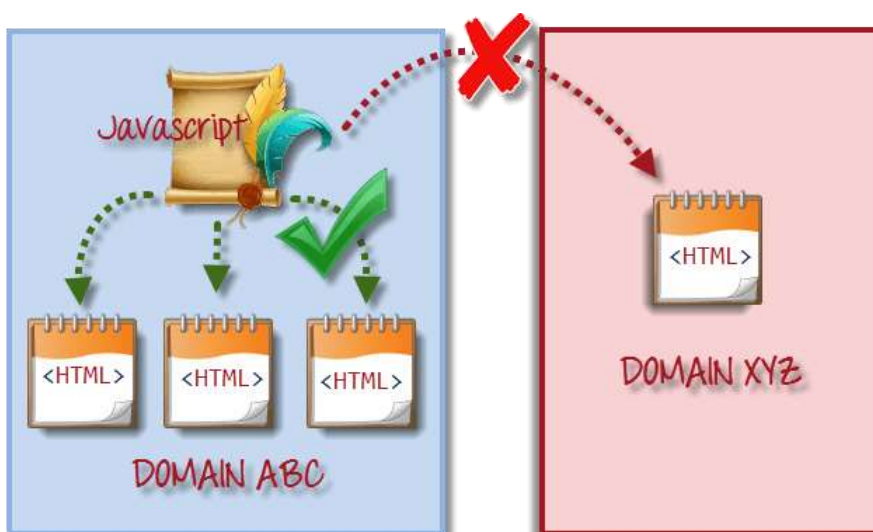


Primarily, Selenium was **created by Jason Huggins in 2004**. An engineer at ThoughtWorks, he was working on a web application that required frequent testing. Having realized that the repetitious Manual Testing of their application was becoming more and more inefficient, he created a JavaScript program that would automatically control the browser's actions. He named this program as the "**JavaScriptTestRunner**."

Seeing potential in this idea to help automate other web applications, he made JavaScriptRunner open-source which was later re-named as **Selenium Core**.

## The Same Origin Policy Issue

**Same Origin policy prohibits JavaScript code from accessing elements from a domain that is different from where it was launched.** Example, the HTML code in `www.google.com` uses a JavaScript program "randomScript.js". The same origin policy will only allow randomScript.js to access pages within google.com such as `google.com/mail`, `google.com/login`, or `google.com/signup`. However, it cannot access pages from different sites such as `yahoo.com/search` or `guru99.com` because they belong to different domains.



*under Same Origin Policy, a Javascript program can only access pages on the same domain where it belongs. it cannot access pages from different domains*

This is the reason why prior to Selenium RC, testers needed to install local copies of both Selenium Core (a JavaScript program) and the web server containing the web application being tested so they would belong to the same domain

## Birth of Selenium Remote Control (Selenium RC)



Paul Hammant

Unfortunately; testers using Selenium Core had to install the whole application under test and the web server on their own local computers because of the restrictions imposed by the **same origin policy**. So another ThoughtWork's engineer, **Paul Hammant**, decided to create a server that will act as an HTTP proxy to "trick" the browser into believing that Selenium Core and the web application being tested come from the same domain. This system became known as the **Selenium Remote Control** or **Selenium 1**.

## Birth of Selenium Grid



Patrick Lightbody

Selenium Grid was developed by **Patrick Lightbody** to address the need of minimizing test execution times as much as possible. He initially called the system "**Hosted QA**." It was capable of capturing browser screenshots during significant stages, and also of **sending out Selenium commands to different machines simultaneously**.

## Birth of Selenium IDE



**Shinya Kasatani** of Japan created **Selenium IDE**, a Firefox extension that can automate the browser through a record-and-playback feature. He came up with this



idea to further increase the speed in creating test cases. He donated Selenium IDE to the Selenium Project in **2006**.

## Birth of WebDriver



**Simon Stewart** created WebDriver circa **2006** when browsers and web applications were becoming more powerful and more restrictive with JavaScript programs like Selenium Core. **It was the first cross-platform testing framework that could control the browser from the OS level.**

## Birth of Selenium 2

In **2008**, the whole Selenium Team decided to merge WebDriver and Selenium RC to form a more powerful tool called **Selenium 2**, with **WebDriver being the core**.

Currently, Selenium RC is still being developed but only in maintenance mode. Most of the Selenium Project's efforts are now focused on Selenium 2.

## So, Why the Name Selenium?

It came from a joke which Jason cracked one time to his team. Another automated testing framework was popular during Selenium's development, and it was by the company called **Mercury Interactive** (yes, the company who originally made QTP before it was acquired by HP). Since Selenium is a well-known antidote for Mercury poisoning, Jason suggested that name. His teammates took it, and so that is how we got to call this framework up to the present.

How selenium got its name:  
Selenium removes  
Mercury from the body



### Brief Introduction 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.



#### 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.

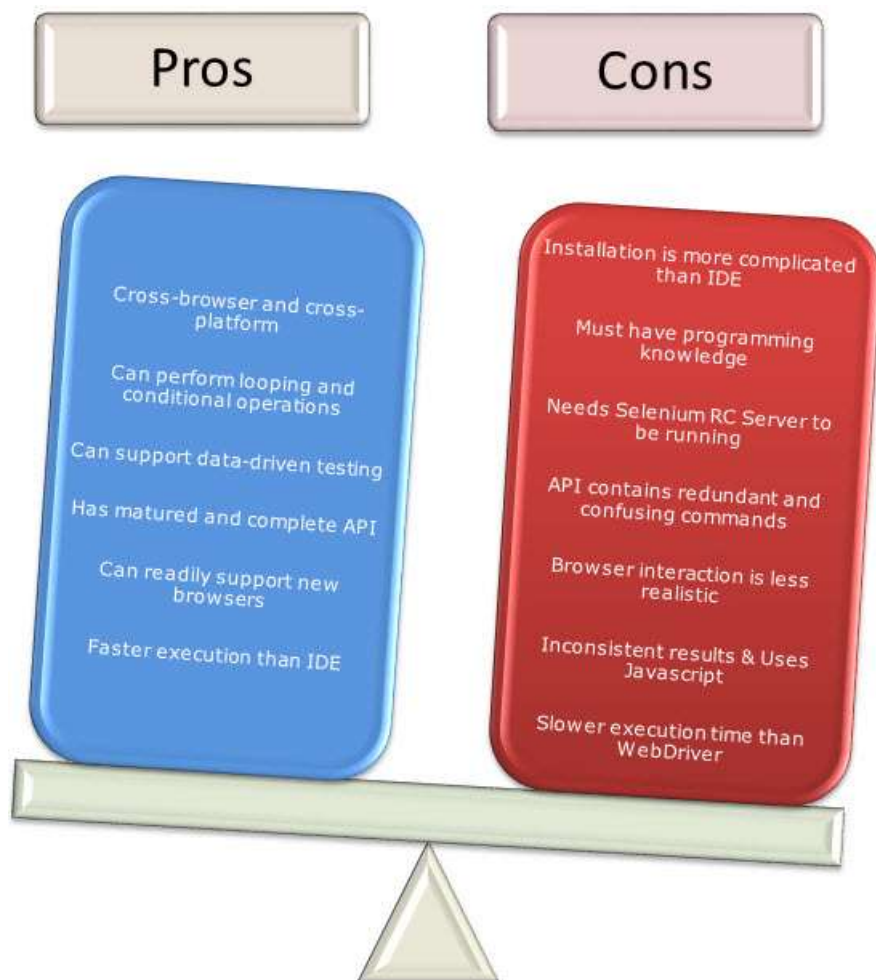
#### CONS

- 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.

## Brief Introduction Selenium Remote Control (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



## 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.

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

Pros	Cons
Simpler installation than Selenium RC	Installation is more complicated than Selenium IDE
Communicates directly to the browser	Requires programming knowledge
Browser interaction is more realistic	Cannot readily support new browsers
No need for a separate component such as the RC Server	Has no built-in mechanism for logging runtime messages and generating test results
Faster execution time than IDE and RC	

## 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.

## Note on Browser and Environment Support

Because of their architectural differences, Selenium IDE, Selenium RC, and WebDriver support different sets of browsers and operating environments.

	Selenium IDE	WebDriver
<b>Browser Support</b>	Mozilla Firefox	Internet Explorer versions 6 to 11, both 32 and 64-bit  Microsoft Edge version 12.10240 & above ( partial support some functionalities under development) Firefox 3.0 and above Google Chrome 12.0. and above Opera 11.5 and above Android - 2.3 and above for phones and tablets (devices & emulators)  iOS 3+ for phones (devices & emulators) and 3.2+ for tablets (devices & emulators)  HtmlUnit 2.9 and above
<b>Operating System</b>	Windows, Mac OS X, Linux	All operating systems where the browsers above can run.

**Note:** Selenium WebDriver is termed as the successor of Selenium RC which has been deprecated & officially announced by SeleniumHQ.

## How to Choose the Right Selenium Tool for Your Need

Tool	Why Choose?
<b>Selenium IDE</b>	<ul style="list-style-type: none"><li>• To learn about concepts on automated testing and Selenium, including:</li><li>• Selenese commands such as type, open, clickAndWait, assert, verify, etc.</li><li>• Locators such as id, name, xpath, css selector, etc.</li><li>• Executing customized JavaScript code using runScript</li><li>• Exporting test cases in various formats.</li><li>• To create tests with little or no prior knowledge in programming.</li><li>• To create simple test cases and test suites that you can export later to RC or WebDriver.</li><li>• To test a web application against Firefox only.</li></ul>

Tool	Why Choose?
<b>Selenium RC</b>	<ul style="list-style-type: none"> <li>• To design a test using a more expressive language than Selenese</li> <li>• To run your test against different browsers (except HtmlUnit) on different operating systems.</li> <li>• To deploy your tests across multiple environments using Selenium Grid.</li> <li>• To test your application against a new browser that supports JavaScript.</li> <li>• To test web applications with complex AJAX-based scenarios.</li> </ul>
<b>WebDriver</b>	<ul style="list-style-type: none"> <li>• To use a certain programming language in designing your test case.</li> <li>• To test applications that are rich in AJAX-based functionalities.</li> <li>• To execute tests on the HtmlUnit browser.</li> <li>• To create customized test results.</li> </ul>
<b>Selenium Grid</b>	<ul style="list-style-type: none"> <li>• To run your Selenium RC scripts in multiple browsers and operating systems simultaneously.</li> <li>• To run a huge test suite, that needs to complete in the soonest time possible.</li> </ul>

## A Comparison between Selenium and QTP(now 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**. The Selenium Tool Suite has many advantages over QTP as detailed below -

Advantages of Selenium over QTP

Selenium	QTP
<b>Open source, free to use, and free of charge.</b>	<b>Commercial.</b>
<b>Highly extensible</b>	Limited add-ons
Can run tests across <b>different browsers</b>	Can only run tests in <b>Firefox, Internet Explorer and Chrome</b>
Supports <b>various operating systems</b>	Can only be used in <b>Windows</b>

Supports <b>mobile devices</b>	QTP Supports Mobile app test automation (iOS & Android) using HP solution called - HP Mobile Center
Can execute tests <b>while the browser is minimized</b>	Needs to have the application under test to be visible on the desktop
Can execute tests <b>in parallel.</b>	Can only execute in parallel but using Quality Center which is again a paid product.

## Advantages of QTP over Selenium

QTP	Selenium
Can test <b>both web and desktop applications</b>	Can only test web applications
Comes with a <b>built-in object repository</b>	Has no built-in object repository
<b>Automates faster than Selenium</b> 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 <b>it has built-in global and local data tables.</b>	Data-driven testing is more cumbersome since you have to rely on the programming language's capabilities for setting values for your test data
<b>Can access controls within the browser</b> (such as the Favorites bar, Address bar, Back and Forward buttons, etc.)	Cannot access elements outside of the web application under test
Provides professional <b>customer support</b>	No official user support is being offered.
Has native capability to <b>export test data</b> 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.

Though clearly, QTP has more advanced capabilities, Selenium outweighs QTP in three main areas:

- **Cost**(because Selenium is completely free)
- **Flexibility**(because of a number of programming languages, browsers, and platforms it can support)
- **Parallel testing**(something that QTP is capable of but only with use of Quality Center)

### Summary

- The entire Selenium Tool Suite is comprised of four components:
  - **Selenium IDE**, a Firefox add-on that you can only use in creating relatively simple test cases and test suites.
  - **Selenium Remote Control**, also known as **Selenium 1**, which is the first Selenium tool that allowed users to use programming languages in creating complex tests.
  - **WebDriver**, the newer breakthrough that allows your test scripts to communicate directly to the browser, thereby controlling it from the OS level.
  - **Selenium Grid** is also a tool that is used with Selenium RC to execute parallel tests across different browsers and operating systems.
- Selenium RC and WebDriver was merged to form **Selenium 2**.
- Selenium is more advantageous than QTP in terms of **costs and flexibility**. It also allows you to **run tests in parallel**, unlike in QTP where you are only allowed to run tests sequentially.

**Buy Now \$9.99**