Introduction

What is Automation?

- Simulation of any task using a system or toll is called as automation

Or

- The use of computers & tool to control a particular process in order to increase reliability and efficiency, is also called as automation.

Why automation is required?

- It is a challenge for any company to continuously maintain and improve the quality and efficiency of software systems development.
- Test automation can improve the development process of a software product in many cases, the automation of tests is initially associated with increased effort, but the related benefits will quickly pay-off.
- Test automation allows performing different types of testing efficiently and effectively.
- Automated tests can run fast and frequently, which is cost-effective for software products with a long maintenance life.
- When testing in an agile environment, the ability to quickly update, react to ever-changing software systems and requirement is necessary. New test cases are generated continuously and can be added to existing automation in parallel to the development of the software itself.
- Optimization of speed, efficiency, quality and the decrease of cost.
- Quick return on investment of test automation.

Advantage or benefits of automation

Fast -> Runs tests significantly faster than human users

Repeatable -> Testers can test, how the software reacts after repeated execution of the same operation.

Reusable -> Scripts can be reused on different versions of the software.

Comprehensive -> Testers can build test suites of tests that covers every feature in software

- Improves accuracy thereby reducing human generated errors
- Saves time and money application

Dis-Advantages

- Initial investment is high
- Require additional skills

When we go for automation?

- We go for automation for long term projects

What type of task we automate?

- Repeating task
  - ➔ Regression testing
  - ➔ Smoke testing

Introduction to Selenium

- What is the best tool for me to get my tests automated?
- Is there a cost involved?
- Is it easy to adapt?
- Why selenium?

One of the best answers to all the above questions for automating web based applications is Selenium. Because:

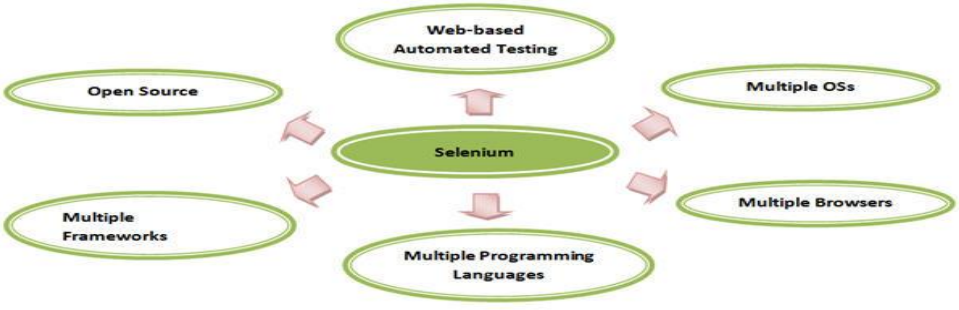
- It's open source
- have a large user base and helping communities
- have multi browser and platform compatibility
- has active repository developments
- supports multiple language implementations

What is Selenium?

Selenium is a free (open source) automated testing suite for web applications across different browsers and platforms. Selenium focuses on automating web-based applications.

Note: Selenium is one of the most popular automated testing suites. Selenium is designed in a way to support and encourage automation testing of functional aspects of web based applications and a wide range of browsers and platforms. Due to its existence in the open source community, it has become one of the most accepted tools amongst the testing professionals.

Selenium supports a broad range of browsers, technologies and platforms.



Selenium supports a broad range of browsers, technologies and platforms

Brief Introduction to Selenium tools

Selenium Core

Selenium is a result of continuous efforts by an engineer at ThoughtWorks, named as Jason Huggins. Being responsible for the testing of an internal Time and Expenses application, he realized the need for an automation testing tool so as to get rid of repetitive manual tasks without compromising with the quality and accuracy.

As a result, he built a JavaScript program, named as "JavaScriptTestRunner" in early 2004 that could automatically control the browser's actions which seemed very much similar to that of a user communicating with the browser.

Henceforth, Jason started demoing the tool to the vast audience. Eventually the discussions were laid out to categorize this tool in the open source category as well as its potential to grow as a re-usable testing framework for other web based applications.

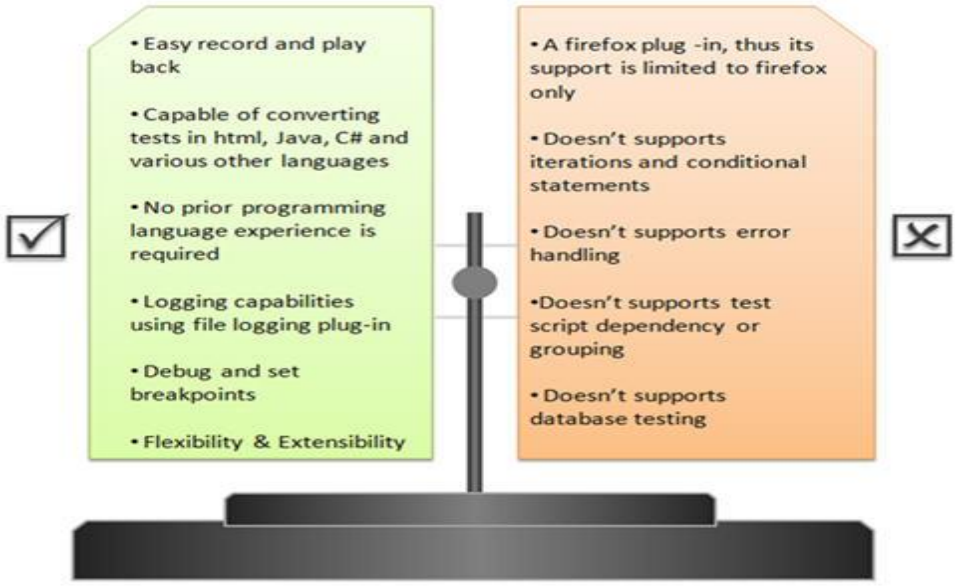
The tool was later on acclaimed with the name "Selenium Core".

Selenium IDE (Selenium Integrated Development Environment)

Selenium IDE was developed by Shinya Kasatani. While studying Selenium Core, he realized that this JavaScript code can be extended to create an integrated development environment (IDE) which can be plugged into Mozilla Firefox. This IDE was capable of recording and playing back the user actions on a Firefox instance to which it was plugged-in. Later on Selenium IDE became a part of Selenium Package in the year 2006. The tool turned out a great value and potential to the community.

Selenium IDE is the simplest and easiest of all the tools within the Selenium Package. Its record and playback feature makes it exceptionally easy to learn with minimal acquaintances to any programming language. With several advantages, a few disadvantages accompanied Selenium IDE, thus making it inappropriate to be used in cases of more advanced test scripts.

Advantages and disadvantages of Selenium IDE:



Advantages and disadvantages of Selenium IDE - © www.SoftwareTestingHelp.com

The disadvantages of IDE are in reality not disadvantages of selenium, rather just limitations to what IDE could achieve. These limitations can be overcome by using Selenium RC or WebDriver.

Selenium RC (Selenium Remote Control)

Paul Hammant

Selenium RC is a tool which is written in java that allows a user to construct test scripts for a web based application in which ever programming language he/she chooses. Selenium RC came as result to overcome various disadvantages incurred by Selenium IDE or Core.

Loopholes and restrictions which were imposed while using Selenium Core made it difficult for the user to leverage the benefits of the tool to its totality. Thus it made the testing process a cumbersome and a far reaching task.

One of the crucial restrictions was **same origin policy.**

Problem of same origin policy:
The problem of same origin policy disallows to access the DOM of a document from an origin that is different from the origin we are trying to access the document.

Origin is a sequential combination of scheme, host and port of the URL. For example, for a URL http://www.seleniumhq.org/projects/, the origin is a combination of http, seleniumhq.org, 80 correspondingly.

Thus the Selenium Core (JavaScript Program) cannot access the elements from an origin that is different from where it was launched.

For Example, if I have launched the JavaScript Program from "http://www.seleniumhq.org/", then I would be able to access the pages within the same domain such as "http://www.seleniumhq.org/projects/" or "http://www.seleniumhq.org/download/". The other domains like google.com, yahoo.com would no more be accessible.
Thus, to test the application using Selenium Core, one has to install the entire application on the Selenium Core as well as web server to overcome the problem of same origin policy.

So, In order to govern the same origin policy without the need of making a separate copy of Application under test on the Selenium Core, Selenium Remote Control was introduced. While Jason Huggins was demoing Selenium, another fellow colleague at ThoughtWorks named Paul Hammant suggested a work around of same origin policy and a tool that can be wired up with a programming language of our choice. Thus Selenium RC came into existence.

Selenium Grid


Patrick Lightbody

With selenium RC, life of a tester has always been positive and favorable until the emerging trends raised a demand to execute same or different test scripts on multiple platforms and browsers concurrently so as to achieve distributed test execution, testing under different environments and saving execution time remarkably. Thus, catering these requirements selenium grid was brought into the picture.

Selenium Grid was introduced by Pat Light body in order to address the need for executing the test suites on multiple platforms simultaneously.

Selenium WebDriver


Simon Stewart

Selenium WebDriver was created by yet another engineer at ThoughtWorks named as Simon Stewart in the year 2006. WebDriver is also a web-based testing tool with a subtle difference with Selenium RC. Since, the tool was built on the fundamental where an isolated client was created for each of the web browser; no JavaScript Heavy lifting was required. This led to a compatibility analysis between Selenium RC and WebDriver. As a result a more powerful automated testing tool was developed called ***Selenium 2***.
WebDriver is clean and a purely object oriented framework. It utilizes the browser's native compatibility to automation without using any peripheral entity. With the increasing demand it has gained a large popularity and user base.

Selenium 3
Selenium 3 is an advance version of Selenium 2. It is a tool focused for automation of mobile and web applications. Stating that it supports mobile testing, we mean to say that the WebDriver API has been extended to address the needs of mobile application testing.

Environment and Technology Stack
With the advent and addition of each new tool in the selenium suite, environments and technologies became more compatible. Here is an exhaustive list of environments and technologies supported by selenium tool set.

Supported Browsers

Supported Programing Languages

Language	Client Versio
Java	3.8.1
C#	3.8.0
Ruby	3.8.0
Python	3.8.1
Javascript (Node)	3.6.0

Perl download and docs by Gordon Child
Perl 6 by Ahmad M. Zawawi
PHP by Chibimagic (real name unknown?)
PHP by Lukasz Kolczynski
PHP by facebook
PHP by Adam Goucher
PHP by Nearsoft
Haskell by Adam Curtis
Objective-C by Dan Cuellar
Javascript by Adam Christian
Javascript by Jonathan Lipps
Javascript by Camilo Tapia, Vincent Voyer and Christian Bromann
JavaScript Leadfoot by SitePen
R by John Harrison
Dart by Marc Fisher
Tcl by Tobias Koch
Elixir by Nathan Johnson

Supported Operating Systems

- Supports all the major Operating Systems.

Supported Testing Frameworks

- Supports all the Unit Test Frameworks.
- Namely.
    - Py-Test
    - TestNG
    - NUnit
    - Junit
    - RSpec
    - Robot Framework
    - Etc.

Birth of Selenium 2

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

Selenium 3

- It is the most anticipated inclusion in the Selenium suite. Selenium 3 strongly encourages mobile testing.

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.