

Automation Testing

Testing: -

- ❖ Testing is the process of correctness and completeness of application or software with respect to customer requirements.
- ❖ The process of software testing aims not only at finding faults in software but also at finding measures to improve the software in terms of efficiency, accuracy, and usability.
- ❖ It mainly aims to provide quality product to users.

Types

1. Manual Testing:

- Manual testing is testing software manually, i.e., without using any automation tool or any script.
- There are different stages for manual testing to test application such as unit testing, integration testing, system testing, and usability testing etc.
- Testers use different documents such as test plans, test cases, or test scenarios to test software to ensure the completeness of testing.

Manual Testing:



Automation Testing:



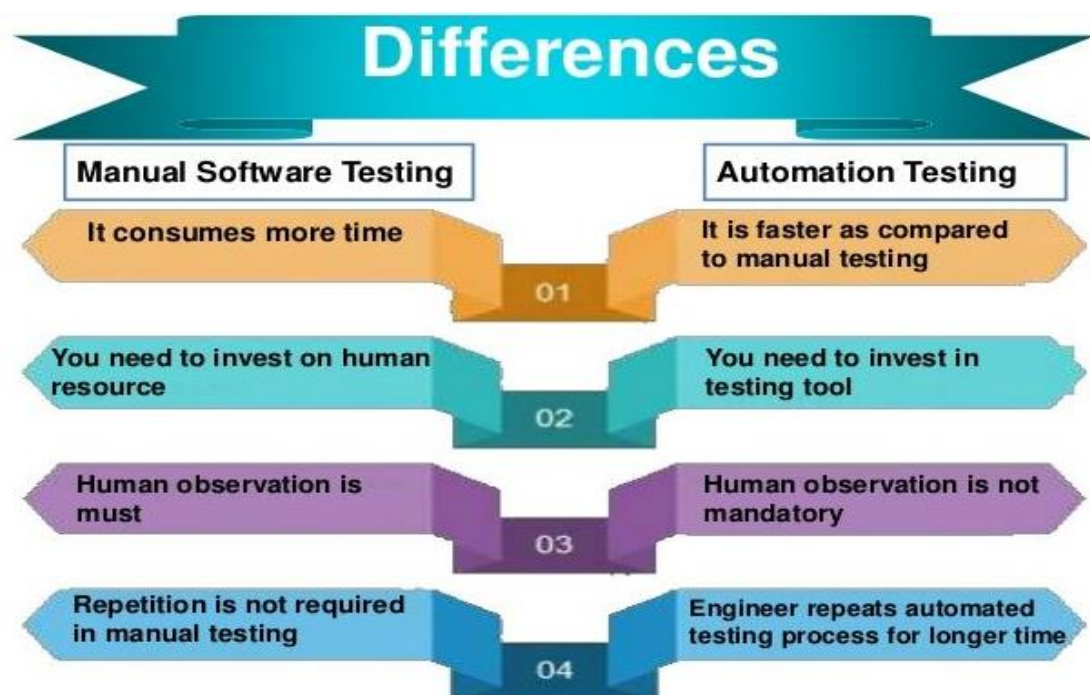
When do you prefer Manual Testing over Automation Testing?

- When the project is in initial development stage.
- When testing user interface especially their visual aspects.
- When exploratory or adhoc testing needs to be performed.

- If the project is a short term and writing scripts will be time consuming when compared to manual testing
- If the test case is not automatable. Example captcha.

2. Automation Testing

- Testing an application or software with the help of an automation tool and executive test script called as automation testing.
- We are comparing actual results and expected results so we can decide our test case is pass or fail.
- Automation Testing is used to re-run the test scenarios quickly and repeatedly, that were performed manually in manual testing.
- Apart from regression testing, automation testing is also used to test the application from a load, performance, and stress point of view.

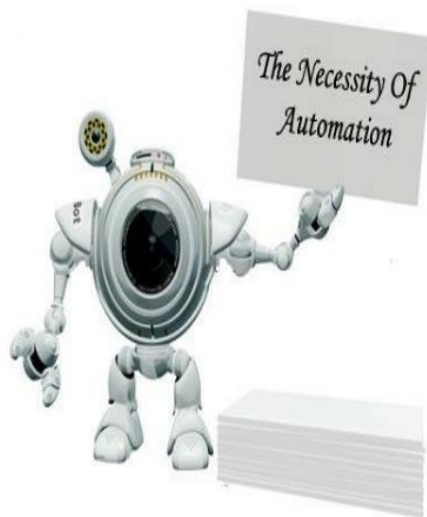


Why we go for automation testing >> Disadvantages of Manual Testing –

- We need to perform retesting and regression testing so it will take **more time and efforts** manual testing.
- More human resources are required in manual testing. (Possibility of **human errors**)

- Accuracy is less
- Need to prepare extra **documentation** for test results.
- **Compatibility testing is difficult** because in which version browser and cross browser compatibility we are going to test so it is difficult in manual testing but it will be easy in automation testing.
- So that why we go for automation.

Why Automated Testing?



Copyright @2016 Janbask.net

- It is difficult to test for multi lingual sites manually.
- Automation does not require Human intervention.
- Automation increases speed of test execution
- Automation helps increase Test Coverage.
- Manual Testing can become boring and hence error prone.
- Manual Testing of all work flows, all fields ,all negative scenarios is time and cost consuming.



info@janbask.net
training@janbask.com

=====

Advantages of Automation Testing

- Reusability of test scripts > We did not write the test cases multiple times so reusability is possible in this we are writing it once and used it many times,
- Project duration reduces > So everything depends on delivery we can reduce the project duration by using agile methodology with automation.
- Compatibility testing is easy > Compatibility testing is easy we can execute it parallely in automation testing.
- Less human efforts
- Accuracy is more

Some of the popular automation testing tools

Tools	Application Under Test	Supported Platforms	Scripting Languages	Pricing
Selenium	Web App	Windows/Mac/ Linux	Java/Python/Ruby/VB.Net	Opensource
Katalon Studio	Web/Mobile/Desktop/ API/Apps	Windows/Mac/ Linux	Java/Groovy	Opensource
UFT One	Web/Desktop/Mobile/RPA Apps	Windows	VBScript	Commercial
Appium	Mobile Apps	Windows/Mobile	Java/JavaScriptP/HP/Ruby/ Python/ C#	Opensource
IBM RFT	Web Apps	Windows/Linux	Java/VB.Net	Commercial
Telerik Test Studio	Web/Mobile Apps	Windows	C#/VB.Net	Commercial
Watir	Web Apps	Windows/Linux	Rubby	Opensource
Test Complete	Web/Desktop/Mobile Apps	Windows	JavaScript/VBScript/Python	Commercial
Cucumber	Web Apps	Windows	Java/Scala/Ruby/ Groovy	Opensource
Ranorex	Web/Desktop/Mobile Apps	Windows	C#/Python/VB.Net	Commercial

Tools :-

- Manual Testing - Without tool
- Automation Testing - Selenium, QTP, Cypress, Katalon etc
- Database Testing - Oracle, MySQL, MangoDB
- API Testing - Postman, SoapUI , Jmeter
- API Automation - Rest Assured
- Performanace Testing - Jmeter, Loadrunner
- Security Testing - Zed Attack Proxy , OWASP
- Mobile testing - Appium

=====

What is Selenium

- Selenium is an open-source framework used to automate the testing of web applications.
- If you are looking forward to automating functional and regression test cases, then Selenium would be the right choice! Test scripts can be written in Selenium using different programming languages like Java, Python, C#, Ruby, and JavaScript.

History of Selenium

- Selenium Invented by Jason Hugins and his colleagues in 2004
- Originally name was JSFT (JavaScript functional tester)
- Built as open-source browser-based integration test framework built by thought works at Chicago for time and expense keeping system
- Developed using JavaScript and HTML
- Designed to make test writing easy
- Ability to step through individual tests

Advantages of selenium >> Selenium: -more demandable

- It is Open source (free)
- It supports Multiple languages: - Java, python, c#, ruby etc...
- It supports Multiple OS: - Windows, MacOS, Linux etc...
- It supports Multiple browsers e.g. chrome, Firefox, safari, opera, edge etc...
- It helps is to do Parallel Testing
- It supports to integrate third party tools like TestNG, Cucumber, Git, Jenkins, GitHub, Maven etc.
- It has very large community



Disadvantages/Limitations of Selenium

- Selenium does not support automation testing for desktop applications.
- Captcha /barcode
- No reporting facility
- File upload

=====

Most common challenges in Automation Testing: -

#1. We cannot test windows application

- Selenium doesn't support windows-based applications. It supports only web-based applications.

#2. We cannot test mobile apps

- Using Selenium testing, we can perform testing on any operating system and browser on the desktop but we can't perform mobile testing with selenium alone. But there is a solution for this.
- You can use Appium to handle iOS and Android native, mobile, and hybrid apps using the WebDriver protocol. Appium allows you to test your application on native mobile operating systems. Appium uses WebDriver protocol to automate mobile app testing instead of web applications.


#3. Limited reporting

- You can't create a decent report using selenium. However, there is a solution.
- You can generate reports using TestNG or Extent reports. These reports will show the information like pass/fail count, execution time, errors etc.,

#4. Handling dynamic Elements

- Some of the web elements are dynamic in nature and aren't immediately visible when you first visit the website.
- If an element's id is changing on every page load then handling these type of elements is bit tricky in the normal way.



- 
- We need to handle the dynamic elements with dynamic xpath or dynamic css selectors. Functions like starts-with, contains, ends with, etc., works well to handle dynamic objects.

#5. Handling page load

- Some of the web pages are user specific. These user-specific pages load different elements depends on the different user. Sometimes some elements appear depends upon the previous action.
- If you choose a country from country dropdown then cities related that country will load in the cities dropdown.
- In runtime selenium script may not identify the element. To overcome this we need to use explicit waits in the script to give the elements enough time to load and to identify the element.

#6. Handling pop up windows

- Windows-based pops are part of the operating system. It's sometimes tough to automate a simple prompt, or confirmation alert pops-up. Selenium does not support native operating system based dialog windows. It's beyond selenium's capabilities.
- We could use AutoIT to handle the windows based popups.

#7. Handling captcha:

- Handling captcha is another challenge in Selenium testing. There are some third-party tools to automate captcha but still, we cannot achieve 100% results.
- We can solve this issue by following below options.
- Completely disable captcha in the test environment
- Make the system to accept a dummy value for captcha in the test environment

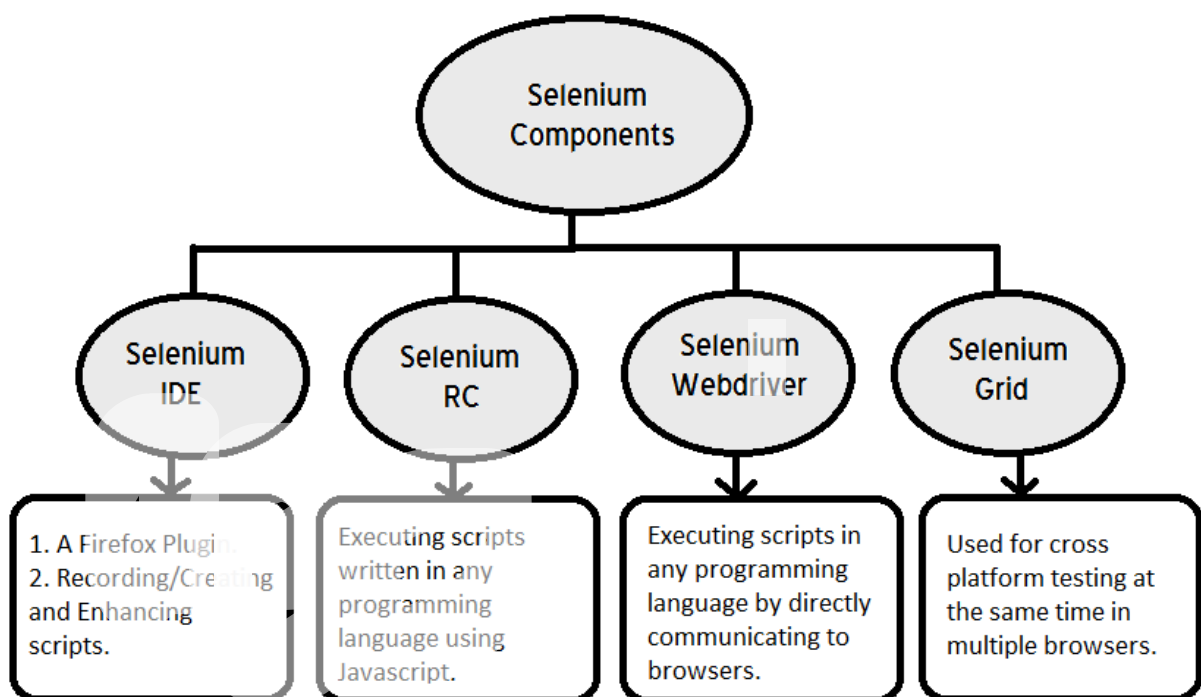
=====



Selenium Components/Selenium Suite

It has four major components which include

1. Selenium IDE (Integrated Development Environment)
2. Selenium RC (Remote Control - Now Deprecated)
3. Selenium WebDriver
4. Selenium Grid



❖ Selenium IDE

Operating System Support – Windows, Mac OS, Linux

Browser Support – Chrome, Firefox and Edge browsers.

- Selenium IDE is plugin/extension which supports Chrome, Firefox and Edge browsers.
- It is a record and playback tools of the automation scripts.
- Also allows testers to export recorded scripts in many languages like HTML, Java, Ruby, RSpec, Python etc.

Disadvantages

- It has limited scope and the generated test scripts are not very robust and portable.
- Data-driven testing is not supported.
- It is not able to test dynamic web applications and Complex operations. So we need to use Selenium RC or Selenium WebDriver to write better scripts.

❖ Selenium RC

Operating System Support – Windows, Mac OS, Linux, Solaris

Browser Support –Firefox, IE, Chrome, Safari, Opera

- Selenium RC AKA Selenium 1. Selenium RC was the main Selenium project for a long time before the WebDriver merge brought up Selenium 2.
- It supports Java, Javascript, Ruby, PHP, Python, Perl and C#.

Disadvantages

- It supports almost every browser out there but doesn't support latest browsers.
- Problems like server down, dependency on server speed slow down.



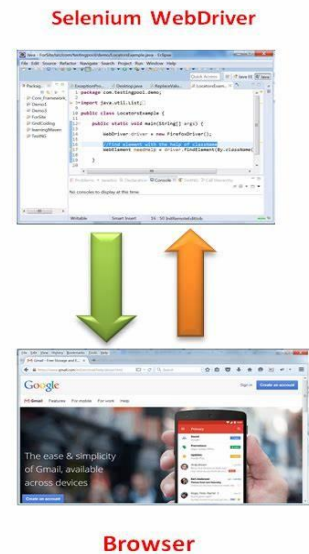
❖ Selenium WebDriver

- Selenium WebDriver AKA Selenium 2 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.

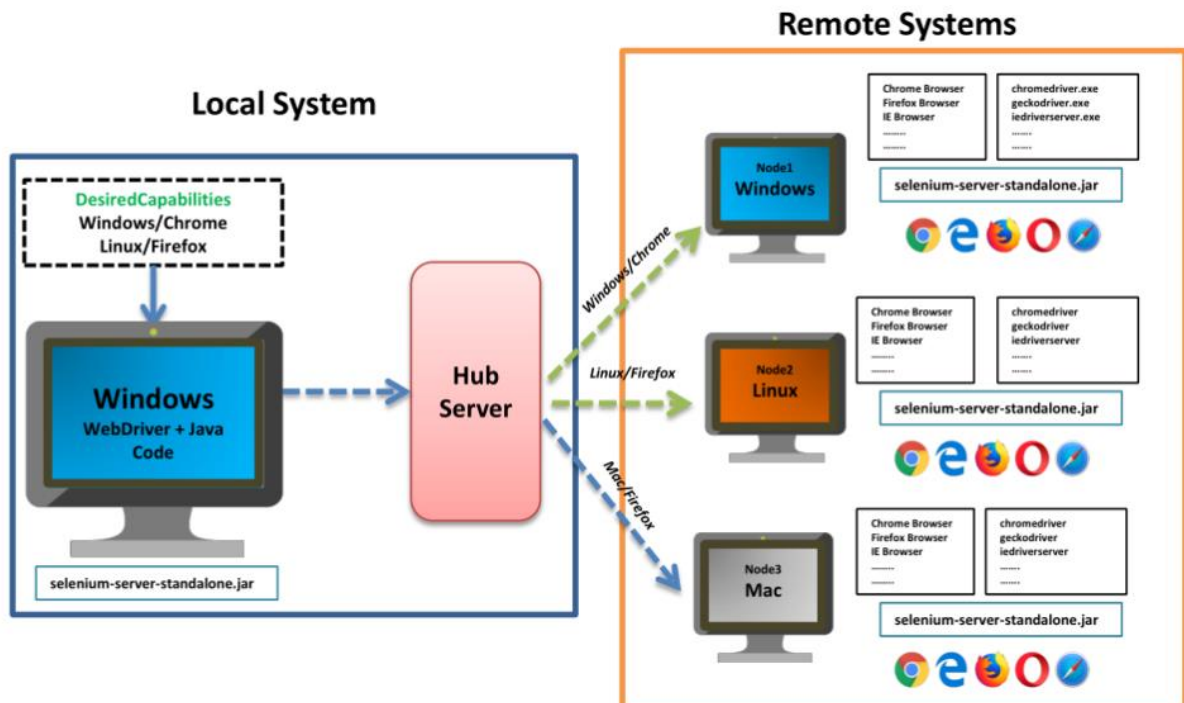
Operation System Support – Windows, Mac OS, Linux, Solaris

Browser Support – Mozilla Firefox, Internet Explorer, Google Chrome 12.0.712.0 and above, Safari, Opera 11.5 and above, Android, iOS etc.

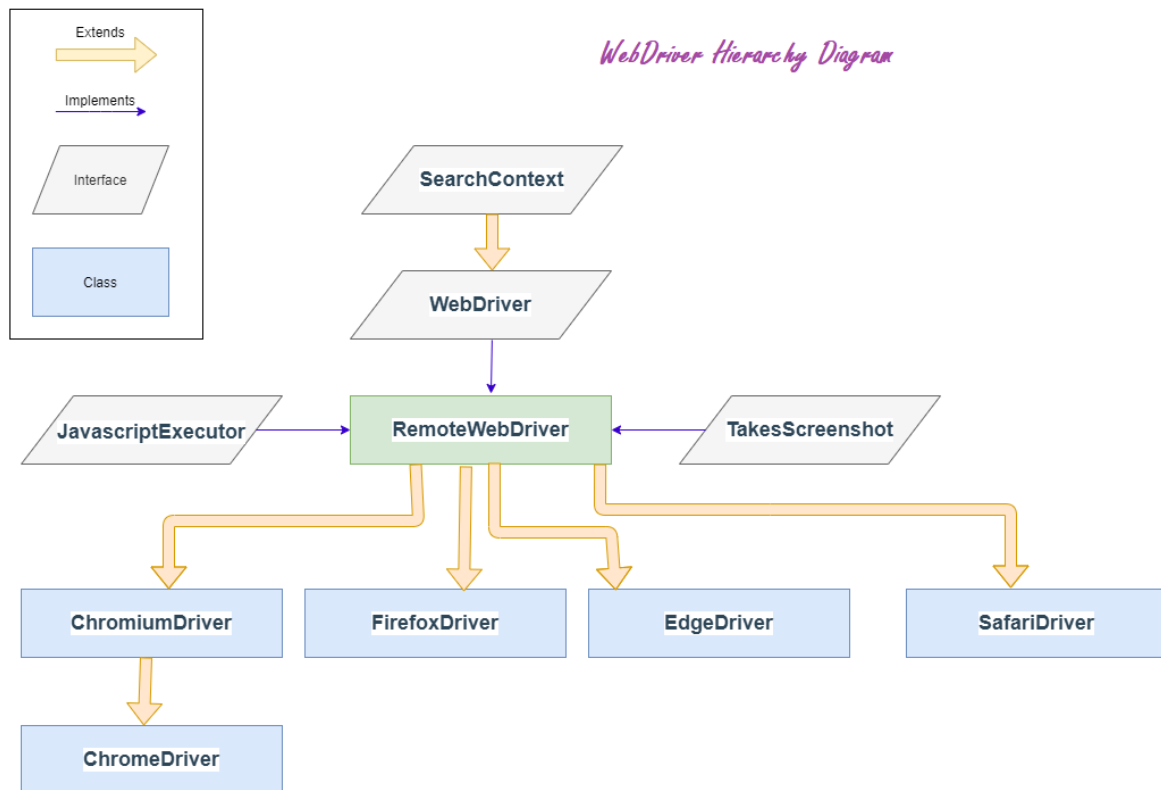


❖ Selenium Grid

- 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 called as remote automations.



WebDriver Hierarchy Diagram



- **SearchContext** is the super most interface in WebDriver hierarchy, which consists of abstract methods and inherited by WebDriver.
 - Methods such as `findElement()` and `findElements`
- **WebDriver** is an Interface which contain abstract methods of both interfaces i.e., search context and its own abstract methods
 - WebDriver interface has multiple inner interfaces which contains methods related to specific events.
- **RemoteWebDriver** is the fully implemented class with access level protected.
- It provides implementation to all the abstract methods from both the interfaces.
 - It also implements other interfaces
 - Implementing `JavascriptExecutor` – helps to scroll, locate elements.
 - Implementing `TakesScreenshot` – helps to capture screenshot.
- Browser such as **Firefox, chrome, safari, edge** etc are subclasses, those are extended to remote WebDriver class.

- Driver classes like ChromeDriver(), EdgeDriver(), FirefoxDriver() etc in hierarchy. These classes provides control on a browser running on the local machine.
- To achieve this, we need to use upcasting in selenium

WebDriver driver = new ChromeDriver();


Selenium Architecture

- **Selenium Client Libraries / Binding Languages** – Selenium can work on various libraries like Java, Python, Ruby, and so on. It has the language bindings for more than one language.
- **JSON Wire Protocol** – JSON is Javascript Object Notion. It is used for transferring data from the server to the client on the web page. It is based on the Rest API that transfers information among HTTP servers.



- **Browser Driver** – All the browsers have a specific browser driver. They interact with the browsers (hiding the logic of browser functionality). As the browser driver receives a command, it is run on the browser and the status of the execution goes back in form of HTTP response.
- **Browser** – Selenium can test applications on browsers like Firefox, Chrome, IE, and so on.

Process –

- 
- Once we trigger our script for execution, the code is converted to an URL via the JSON Wire Protocol over HTTP.
 - The URL will then be fed to the browser driver. The browser driver takes the help of HTTP server to get the HTTP request.
 - The browser driver then passes on the request to the actual browser via HTTP. Finally, the test script gets executed.

=====

