

The background of the slide is a dark, textured surface with a bokeh effect. It features numerous out-of-focus light spots in warm tones of orange, yellow, and red. Interspersed among these circles are various translucent, geometric shapes, including hexagons and pentagons, in similar warm colors. The overall effect is a soft, glowing, and abstract pattern.

Welcome!

Test Automation - Selenium
by
Girish Godbole

Test Automation

- **Why Automation is Important**

- Given the right tools, automating computer operations can be surprisingly easy and can reap major benefits. Understanding these benefits—and some obstacles—will help you develop support for an operations automation project. A recent study by a leading trade journal asked the question, “What do you see as the most important benefits of an automated or unattended computer center?” The primary benefits of operations automation cited most often were **cost reduction, productivity, availability, reliability, and performance.**

1. Reducing Operational Costs
2. Increasing Productivity
3. Ensuring High Availability
4. Increasing Reliability
5. Optimizing Performance
6. Time Saving
7. Enhance workflow efficiency

Automation

- **What is Automation Testing?**
- **Automation Testing** is a software testing technique that performs using special automated testing software tools to execute a test case suite. On the contrary, Manual Testing is performed by a human sitting in front of a computer carefully executing the test steps.
- The automation testing software can also enter test data into the System Under Test, compare expected and actual results and generate detailed test reports. Software Test Automation demands considerable investments of money and resources.

Automation

- **Which Test Cases to Automate?**
- Test cases to be automated can be selected using the following criterion to increase the automation ROI
- High Risk – Business Critical test cases
- Test cases that are repeatedly executed
- Test Cases that are very tedious or difficult to perform manually
- Test Cases which are time-consuming
- **The following category of test cases are not suitable for automation:**
- Test Cases that are newly designed and not executed manually at least once
- Test Cases for which the requirements are frequently changing
- Test cases which are executed on an ad-hoc basis.

Automation

Automated Testing Process:

Following steps are followed in an Automation Process

Step 1) Test Tool Selection

Step 2) Define scope of Automation

Step 3) Planning, Design and Development

Step 4) Test Execution

Step 5) Maintenance



Test Tool Selection

Define scope of Automation

Planning, Design and Development

Test Execution

Maintenance

Automation

- **Test tool selection**
- Test Tool selection largely depends on the technology the Application Under Test is built on. For instance, QTP does not support Informatica. So QTP cannot be used for testing Informatica applications. **It's a good idea to conduct a Proof of Concept of Tool on AUT.**
- **Define the scope of Automation**
- The scope of automation is the area of your Application Under Test which will be automated. Following points help determine scope:
 - The features that are important for the business
 - Scenarios which have **a large amount of data**
 - **Common functionalities** across applications
 - Technical feasibility
 - The extent to which business components are reused
 - **The complexity** of test cases
 - Ability to use the same test cases for cross-browser testing

Automation

- **Framework for Automation**

- A framework is set of automation guidelines which help in
- Maintaining consistency of Testing
- Improves test structuring
- Minimum usage of code
- Less Maintenance of code
- Improve re-usability
- Non Technical testers can be involved in code
- The training period of using the tool can be reduced
- Involves Data wherever appropriate

Automation Frameworks

- 1.Data Driven Automation Framework
- 2.Keyword Driven Automation Framework
- 3.Modular Automation Framework
- 4.Hybrid Automation Framework

Manual Vs Automation

| Parameter | Automation Testing | Manual Testing |
|---------------------|--|---|
| Definition | Automation Testing uses automation tools to execute test cases. | In manual testing, test cases are executed by a human tester and software. |
| Processing time | Automated testing is significantly faster than a manual approach. | Manual testing is time-consuming and takes up human resources. |
| Exploratory Testing | Automation does not allow random testing | Exploratory testing is possible in Manual Testing |
| Initial investment | The initial investment in the automated testing is higher. Though the ROI is better in the long run. | The initial investment in the Manual testing is comparatively lower. ROI is lower compared to Automation testing in the long run. |
| Reliability | Automated testing is a reliable method, as it is performed by tools and scripts. There is no testing Fatigue. | Manual testing is not as accurate because of the possibility of the human errors. |
| UI Change | For even a trivial change in the UI of the AUT, Automated Test Scripts need to be modified to work as expected | Small changes like change in id, class, etc. of a button wouldn't thwart execution of a manual tester. |
| Investment | Investment is required for testing tools as well as automation engineers | Investment is needed for human resources. |

Manual Vs Automation

| Parameter | Automation Testing | Manual Testing |
|------------------------|--|---|
| Cost-effective | Not cost effective for low volume regression | Not cost effective for high volume regression. |
| Test Report Visibility | With automation testing, all stakeholders can login into the automation system and check test execution results | Manual Tests are usually recorded in an Excel or Word, and test results are not readily/ readily available. |
| Human observation | Automated testing does not involve human consideration. So it can never give assurance of user-friendliness and positive experience. | The manual testing method allows human observation, which may be useful to offer user-friendly system. |
| Performance Testing | Performance Tests like Load Testing, Stress Testing, Spike Testing, etc. have to be tested by an automation tool compulsorily. | Performance Testing is not feasible manually |
| Parallel Execution | This testing can be executed on different operating platforms in parallel and reduce test execution time. | Manual tests can be executed in parallel but would need to increase your human resource which is expensive |
| Batch testing | You can Batch multiple Test Scripts for nightly execution. | Manual tests cannot be batched. |
| Programming knowledge | Programming knowledge is a must in automation testing. | No need for programming in Manual Testing. |

Manual Vs Automation

| Parameter | Automation Testing | Manual Testing |
|----------------------------|---|---|
| Set up | Automation test requires less complex test execution set up. | Manual testing needs have a more straightforward test execution setup |
| Engagement | Done by tools. Its accurate and never gets bored! | Repetitive Manual Test Execution can get boring and error-prone. |
| Ideal approach | Automation testing is useful when frequently executing the same set of test cases | Manual testing proves useful when the test case only needs to run once or twice. |
| Build Verification Testing | Automation testing is useful for Build Verification Testing (BVT). | Executing the Build Verification Testing (BVT) is very difficult and time-consuming in manual testing. |
| Deadlines | Automated Tests have zero risks of missing out a pre-decided test. | Manual Testing has a higher risk of missing out the pre-decided test deadline. |
| Framework | Automation testing uses frameworks like Data Drive, Keyword, Hybrid to accelerate the automation process. | Manual Testing does not use frameworks but may use guidelines, checklists, stringent processes to draft certain test cases. |

Manual Vs Automation

| Parameter | Automation Testing | Manual Testing |
|---------------|--|---|
| Documentation | Automated Tests acts as a document provides training value especially for automated unit test cases. A new developer can look into a unit test cases and understand the code base quickly. | Manual Test cases provide no training value |
| Test Design | Automated Unit Tests enforce/drive Test Driven Development Design. | Manual Unit Tests do not drive design into the coding process |
| Devops | Automated Tests help in Build Verification Testing and are an integral part of DevOps Cycle | Manual Testing defeats the automated build principle of DevOps |
| When to Use? | Automated Testing is suited for Regression Testing, Performance Testing, Load Testing or highly repeatable functional test cases. | Manual Testing is suitable for Exploratory, Usability and Adhoc Testing. It should also be used where the AUT changes frequently. |

Manual Testing Pros and Cons

Pros of Manual Testing:

- Get fast and accurate visual feedback
- It is less expensive as you don't need to spend your budget for the automation tools and process
- Human judgment and intuition always benefit the manual element
- While testing a small change, an automation test would require coding which could be time-consuming. While you could test manually on the fly.

Cons of Manual Testing:

- Less reliable testing method because it's conducted by a human. Therefore, it is always prone to mistakes & errors.
- The manual testing process can't be recorded, so it is not possible to reuse the manual test.
- In this testing method, certain tasks are difficult to perform manually which may require an additional time of the software testing phase.

Automated Testing Pros and Cons

Pros of automated testing:

- Automated testing helps you to find more bugs compare to a human tester
- As most of the part of the testing process is automated, you can have a speedy and efficient process
- Automation process can be recorded. This allows you to reuse and execute the same kind of testing operations
- Automated testing is conducted using software tools, so it works without tiring and fatigue unlike humans in manual testing
- It can easily increase productivity because it provides fast & accurate testing result
- Automated testing support various applications
- Testing coverage can be increased because of automation testing tool never forget to check even the smallest unit

Cons of Automated Testing:

- Without human element, it's difficult to get insight into visual aspects of your UI like colors, font, sizes, contrast or button sizes.
- The tools to run automation testing can be expensive, which may increase the cost of the testing project.
- Automation testing tool is not yet foolproof. Every automation tool has their limitations which reduces the scope of automation.
- Debugging the test script is another major issue in the automated testing. Test maintenance is costly.

Why Use Selenium?

Beating the tedious challenges of *manual testing* is the obvious reason behind using automated tools.

And, here's why Selenium is the best choice:



Being **Open-Source**, Selenium is available for **free**.



Selenium works good with **any Operating System**.



Selenium lets you perform tests on **any Web Browser**.



Selenium supports multiple **Programming Languages**.

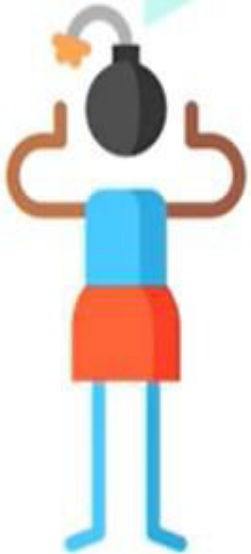


What is Selenium?

“Selenium is an open-source tool that can automate almost any web browser.”



So many test cases, so much work..... So much stress !!!!!



Testing with Selenium has made life so relaxing !!



- Selenium can only be used to test web applications.
- Selenium is fast and easy to use even with large sets of data, and has a guaranteed accuracy.
- Selenium directly runs scripts for any web browser to automate the web application and test it.

Selenium vs. its Counterparts

| Features | HP QTP | IBM RFT | TestComplete | Selenium |
|---------------------|--------------|--------------|--|------------------------------------|
| License | Required | Required | Required | Open Source |
| Cost | High | High | High | Free |
| Customer support | Yes | Yes | Yes | Yes; Open source community |
| Coding skills | Low | Low | High | Very High |
| Environment support | Only Windows | Only Windows | Windows only (7, Vista, Server 2008 or later OS) | Windows, Linux, Mac |
| Language support | VB Script | Java and C# | VB Script, JS Script, Delphi Script, C++ & C# | Java, C#, Ruby, Python, Perl & PHP |

Versions and Suite of Tools



How to Set-up Selenium?



How to Set-up Selenium?

The following three software are prerequisite to begin using Selenium.

I. Java → *Programming Language to write scripts*



II. Eclipse → *Environment to compile and run scripts*



III. Selenium → *Framework for testing web applications*



Selenium Architecture

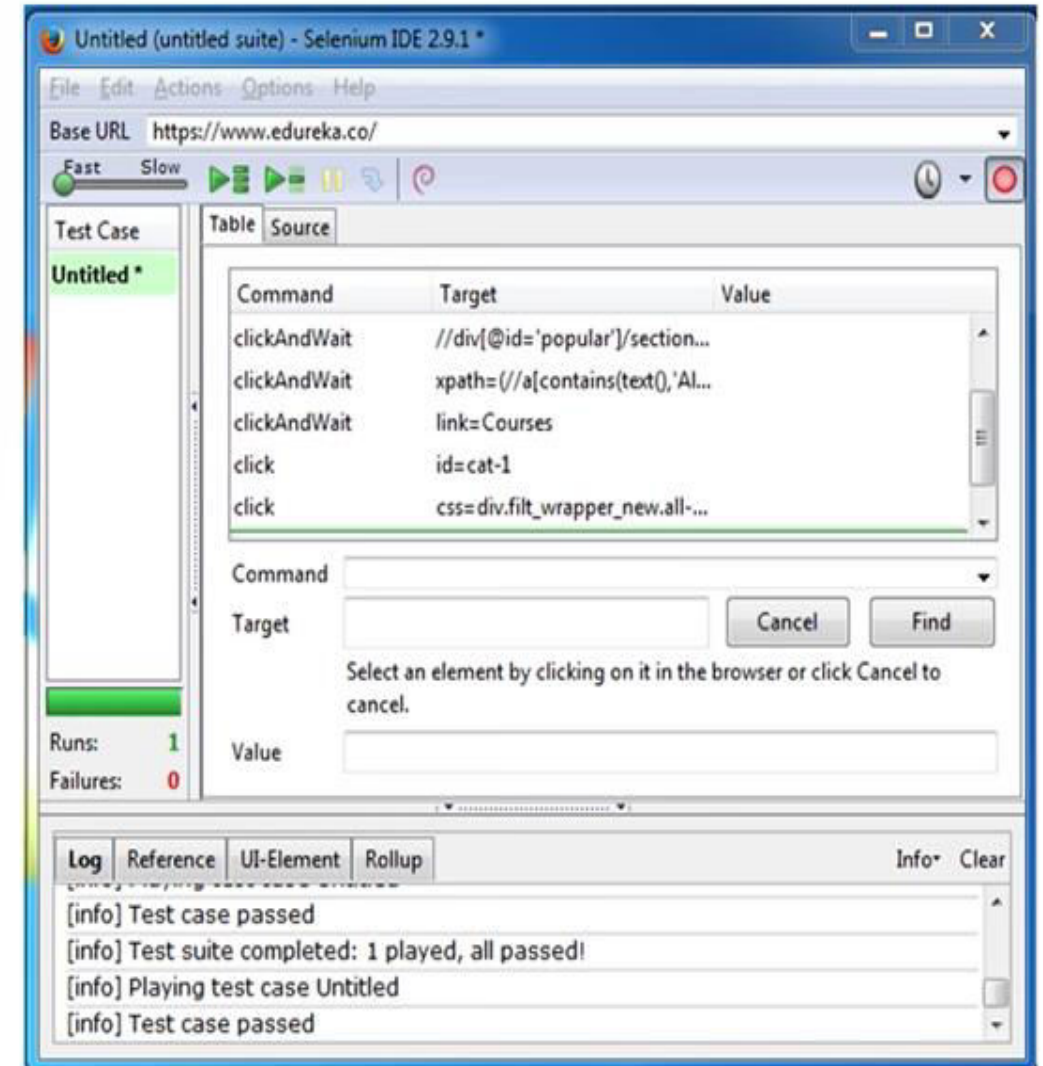


What is Selenium IDE



Selenium IDE

- Selenium-IDE (Integrated Development Environment), a tool to develop test cases
- It is an easy-to-use Firefox plug-in
- You can create and edit test cases and test suites
- You can record and playback tests
- You can convert test cases to different languages



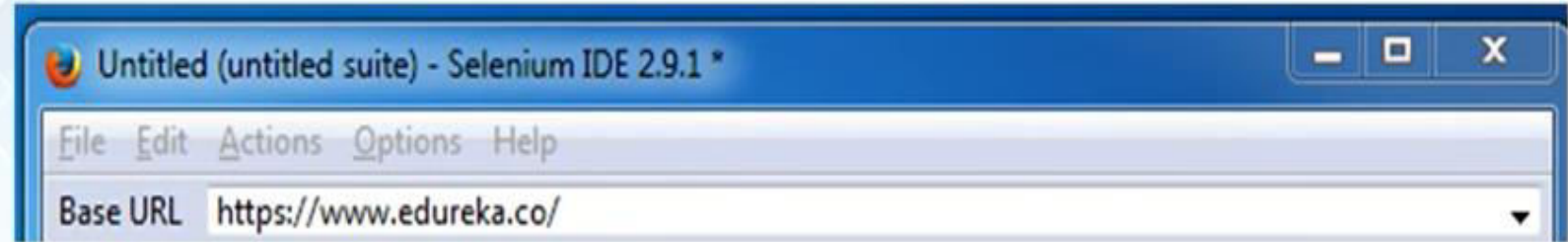
Selenium IDE - Features

Menu Bar

Toolbar

Test Case Pane

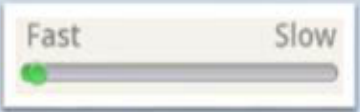






Log/ Reference



- **File** – Provides options to open, save and export test case and Test Suite
- **Edit** – Provides copy, paste, delete, undo and select all operations for editing
- **Actions** – Provides options for record and play
- **Options** – gives options to change IDE settings
- **Help** – Provides links for blogs and documentation

Selenium IDE - Features



| Icon | Name | Description |
|---|--------------------|--|
|  | Speed Control | Control how fast your test case runs |
|  | Run All | Runs the entire Test Suite |
|  | Run | Runs the currently selected test |
|  | Pause/Resume | Allows starting and re-starting of a running test case |
|  | Step | "Step" through test case by running it one command at a time |
|  | Apply Rollup Rules | Repetitive sequences of Selenium commands |
|  | Record | Records the user's browser action |

Menu Bar

Toolbar

Test Case Pane

Log/ Reference

Selenium IDE - Features

Menu Bar

Toolbar

Test Case Pane

Log/ Reference

| Table Source | | |
|--------------|-----------------------------------|-------|
| Command | Target | Value |
| open | / | |
| clickAndWait | css=img[alt="edureka!"] | |
| clickAndWait | //div[@id='popular']/section... | |
| clickAndWait | xpath=//a[contains(text(),'Al...] | |
| clickAndWait | link= Courses | |

| | | |
|---|----------------------|---|
| Command | <input type="text"/> | |
| Target | <input type="text"/> | <input type="button" value="Cancel"/> <input type="button" value="Find"/> |
| Select an element by clicking on it in the browser or click Cancel to cancel. | | |
| Value | <input type="text"/> | |

➤ Has 2 tabs:

- **Table** – Displays the command and their parameters in a readable table format
- **Source** – Displays the test case in the native format in which the file will be stored

➤ The **Command**, **Target** and **Value** entry fields display the currently selected command along with its parameters

Building Test - IDE



Recording

- Begin by recording a test-case from interactions with a website
- Automatic insertion of commands based on your actions
- Typically, includes:
 - Click or clickAndWait
 - Type command
 - Select command
 - Click command



Verification And Asserts

- You need to check the properties of a web-page
- This requires assert and verify commands



Editing

- Insert Command
- Insert Comment
- Edit a Command or Comment

Running Test Cases - IDE



Run a Test Case

- Click the run button
- Runs the currently displayed test case



Run a Test Suite

- Click the run all button
- Runs all the test cases in the currently loaded test suite



Stop and Start

- Pause button can be used to stop the test case while it is running
- The icon then changes to indicate the Resume button



Stop in the middle

- Set breakpoints in the test case to cause it to stop on a particular command



Start from the middle

- Tell the IDE to begin running from a specific command in the middle of the test case



Run any single command

- Double-click any single command to run it by itself

What you cant do using IDE

- ✓ Cannot be used to test big data
- ✓ Recording all possible cases for automation is not possible like file upload, screen shots, etc
- ✓ Cannot handle dynamic part of web application
- ✓ Cannot test connections with database
- ✓ Cannot handle multiple windows



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).



Selenium WebDriver

WebDriver is a tool for testing web applications **across different browsers** using different programming languages. You are now able to make powerful tests because WebDriver **allows you to use a programming language** of your choice in designing your tests.

WebDriver is **faster than Selenium RC** because of its simpler architecture.

WebDriver **directly talks to the browser** while Selenium RC needs the help of the RC Server in order to do so.

WebDriver's API is more **concise** than Selenium RC's.

WebDriver **can support HtmlUnit** while Selenium RC cannot.

The only drawbacks of WebDriver are:

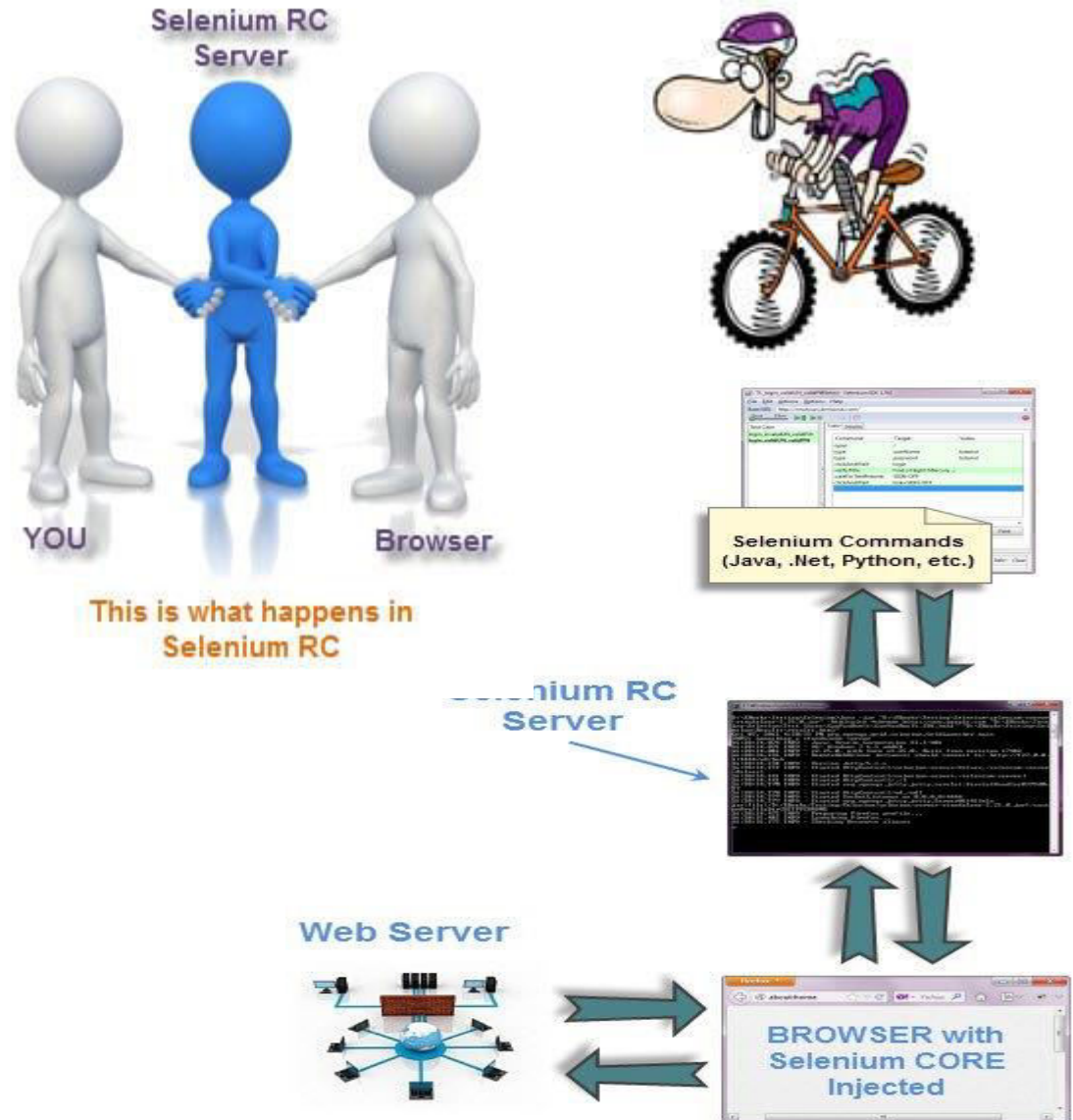
- It **cannot readily support new browsers**, but Selenium RC can.
- It **does not have a built-in command** for automatic generation of test results.

Selenium RC Vs WebDriver

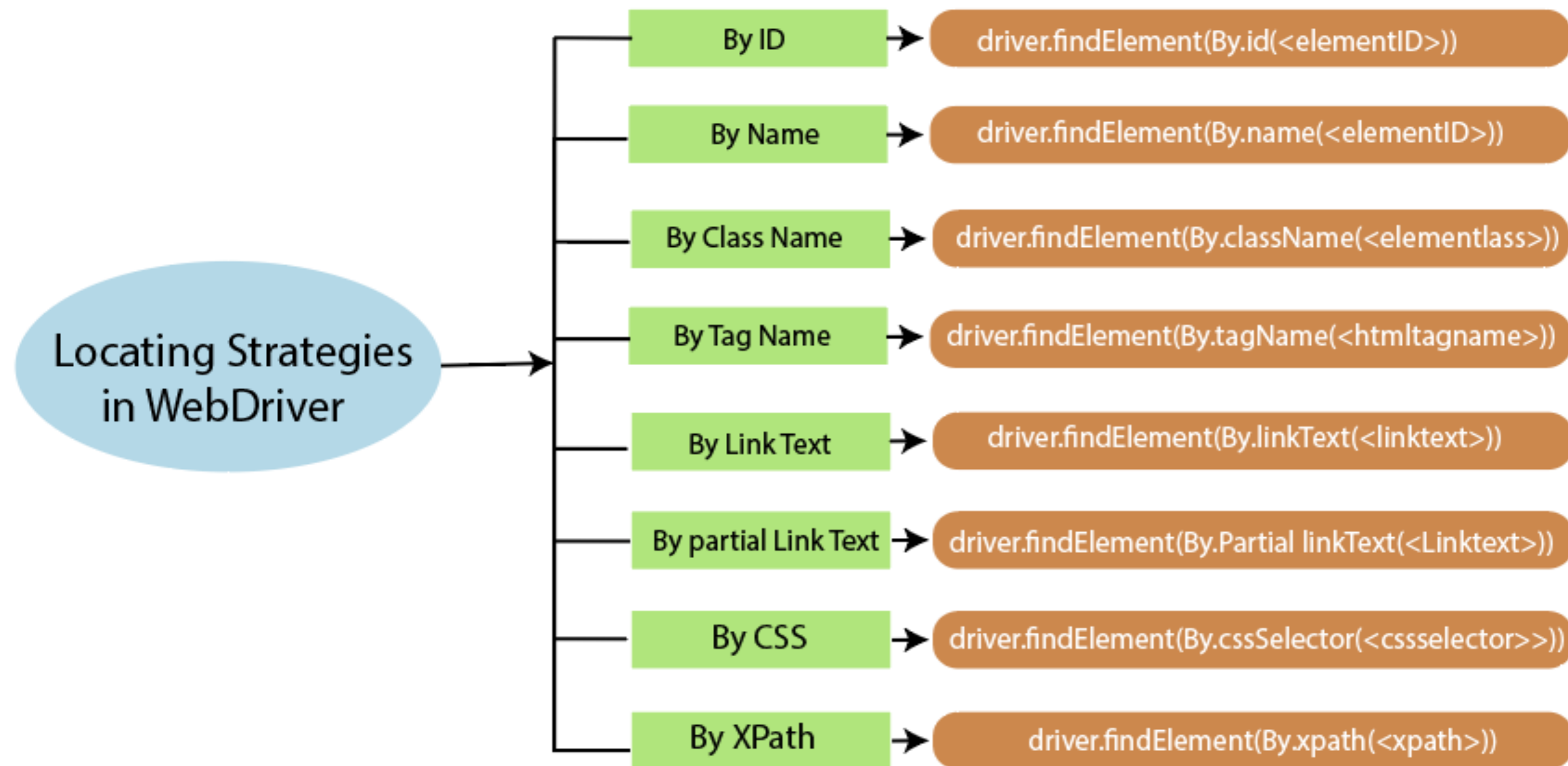
WebDriver



RC – Remote Control



WebDriver Locators



IDE vs RC vs WebDriver

| IDE | RC | WebDriver |
|--|--|--|
| It only works in Mozilla browser. | It supports with all browsers like Firefox, IE, Chrome, Safari, Opera etc. | It supports with all browsers like Firefox, IE, Chrome, Safari, Opera etc. |
| It supports Record and playback | It doesn't supports Record and playback | It doesn't supports Record and playback |
| Doesn't required to start server before executing the test script. | Required to start server before executing the test script. | Doesn't required to start server before executing the test script. |
| It is a GUI Plug-in | It is standalone java program which allow you to run Html test suites. | It actual core API which has binding in a range of languages. |
| Core engine is Javascript based | Core engine is JavaScript based | Interacts natively with browser application |
| Very simple to use as it is record & playback. | It is easy and small API | As compared to RC, it is bit complex and large API. |
| It is not object oriented | API's are less Object oriented | API's are entirely Object oriented |
| It doesn't supports of moving mouse cursors. | It doesn't supports of moving mouse cursors. | It supports of moving mouse cursors. |
| Need to append full xpath with 'xpath=\\' syntax | Need to append full xpath with 'xpath=\\' syntax | No need to append full xpath with 'xpath=\\' syntax |
| It does not supports listeners | It does not supports listeners | It supports the implementation of listeners |
| It does not support to test iphone/Android applications | It does not support to test iphone/Android applications | It support to test iphone/Android applications |

WebDriver Elements -Methods

Basic Methods in WebDriver Interface

- `get(java.lang.String url)`
 - Load a new web page in the current browser window.
- `manage()`
 - Gets the Option interface.
- `getCurrentUrl()`
 - Get a string representing the current URL that the browser is looking at.
- `getTitle()`
 - The title of the current page.
- `getPageSource()`
 - Get the source of the last loaded page.
- `navigate()`
 - An abstraction allowing the driver to access the browser's history and to navigate to a given URL.
- `quit()`
 - Quits this driver, closing every associated window.
- `close()`
 - Close the current window, quitting the browser if it's the last window currently open.

WebDriver Elements -Methods

Basic Methods in WebDriver Interface

- **getWindowHandle()**
 - Return an opaque handle to this window that uniquely identifies it within this driver instance.
- **getWindowHandles()**
 - Return a set of window handles which can be used to iterate over all open windows of this WebDriver instance by passing them to `switchTo().WebDriver.Options.window()`
- **switchTo()**
 - Send future commands to a different frame or window.
- **findElement(By by)**
 - Find the first WebElement using the given method.
- **findElements(By by)**
 - Find all elements within the current page using the given mechanism.

WebDriver Elements -Methods

Working with WebElements in Selenium WebDriver

- `sendKeys(java.lang.CharSequence... keysToSend)` - Use this method to simulate typing into an element, which may set its value.
- `clear()` - If this element is a text entry element, this will clear the value.
- `click()` - Click this element.
- `getAttribute(java.lang.String name)` - Get the value of the given attribute of the element.
- `getCssValue(java.lang.String propertyName)` - Get the value of a given CSS property.
- `getLocation()` - Where on the page is the top left-hand corner of the rendered element?
- `getSize()` - What is the width and height of the rendered element?
- `getTagName()` - Get the tag name of this element.
- `getText()` - Get the visible text
- `isDisplayed()` - Is this element displayed or not? This method avoids the problem of having to parse an element's "style" attribute.
- `isEnabled()` - Is the element currently enabled or not? This will generally return true for everything but disabled input elements.
- `isSelected()` - Determine whether or not this element is selected or not.

What is Selenium Grid?

Selenium Grid is a part of the Selenium Suite that specializes in running multiple tests across different browsers, operating systems, and machines in parallel. It is achieved by routing the commands of remote browser instances where a server acts as a hub. A user needs to configure the remote server in order to execute the tests.

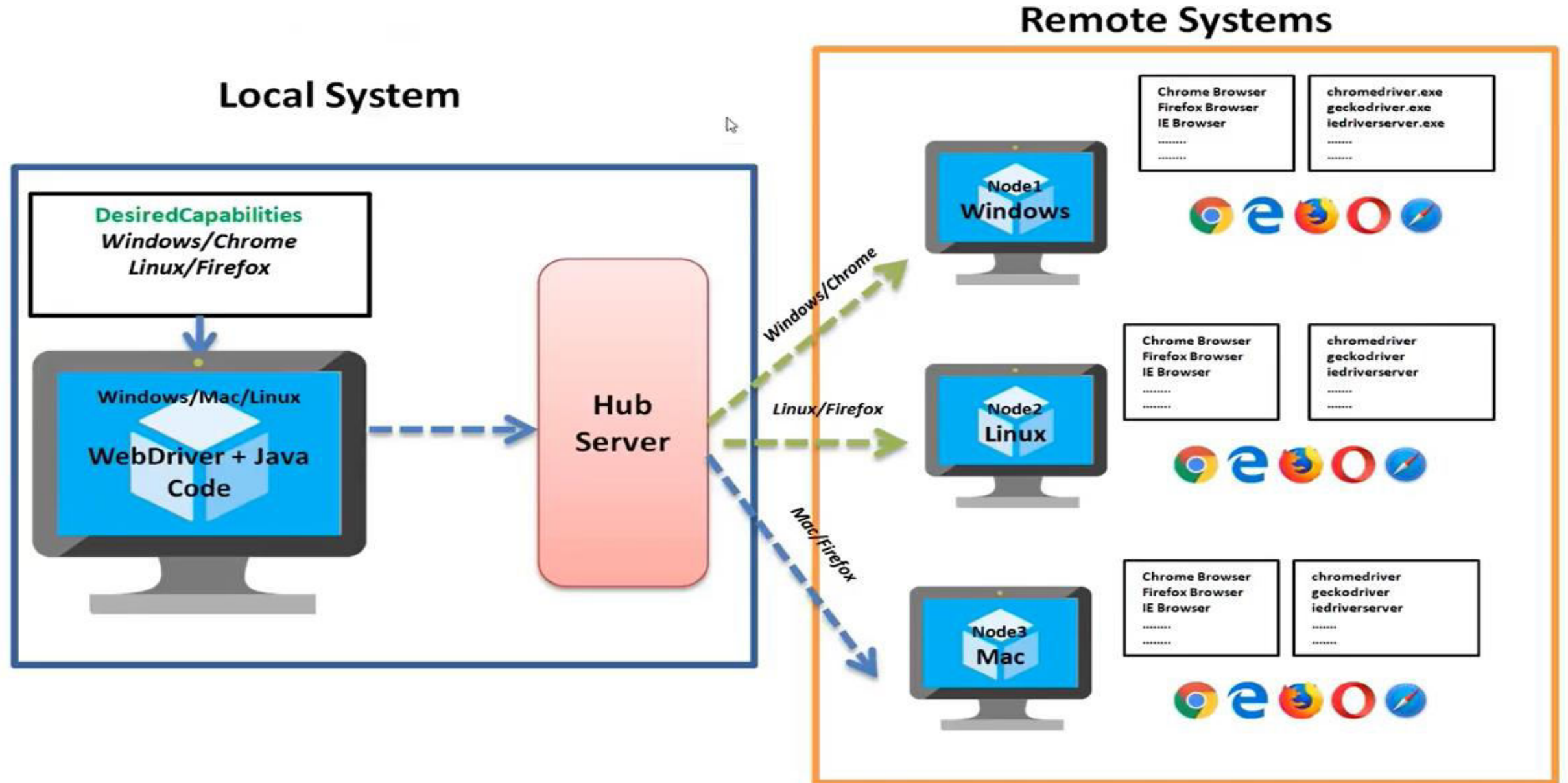
Selenium Grid uses a hub-node concept where you only run the test on a single machine called a **hub**, but the execution will be done by different machines called **nodes**.

When to Use Selenium Grid?

You should use Selenium Grid when you want to do either one or both of following:

- **Run your tests against different browsers, operating systems, and machines all at the same time.** This will ensure that the application you are Testing is fully compatible with a wide range of browser-O.S combinations.
- **Save time in the execution of your test suites.** If you set up Selenium Grid to run, say, 4 tests at a time, then you would be able to finish the whole suite around 4 times faster.

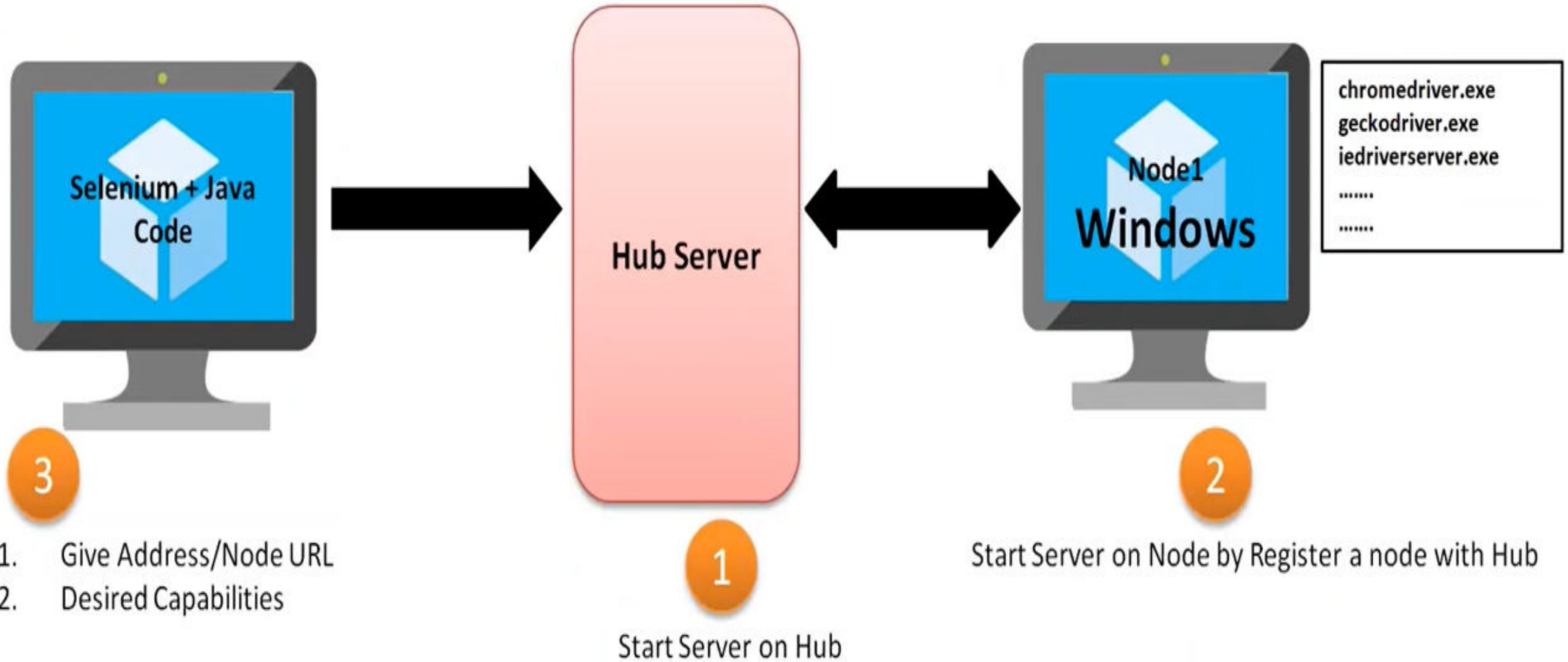
Selenium Grid



Selenium Grid

Local System

Remote System



Selenium Grid Architecture

Selenium Grid has a Hub and Node Architecture.

The Hub

- The hub is the central point where you load your tests into.
- There should only be one hub in a grid.
- The hub is launched only on a single machine, say, a computer whose O.S is Windows 7 and whose browser is IE.
- The machine containing the hub is where the tests will be run, but you will see the browser being automated on the node.

The Nodes

- Nodes are the Selenium instances that will execute the tests that you loaded on the hub.
- There can be one or more nodes in a grid.
- Nodes can be launched on multiple machines with different platforms and browsers.
- The machines running the nodes need not be the same platform as that of the hub.

Selenium grid can be set up in two different ways; one through command line and the other through JSON config file.

Thank You

