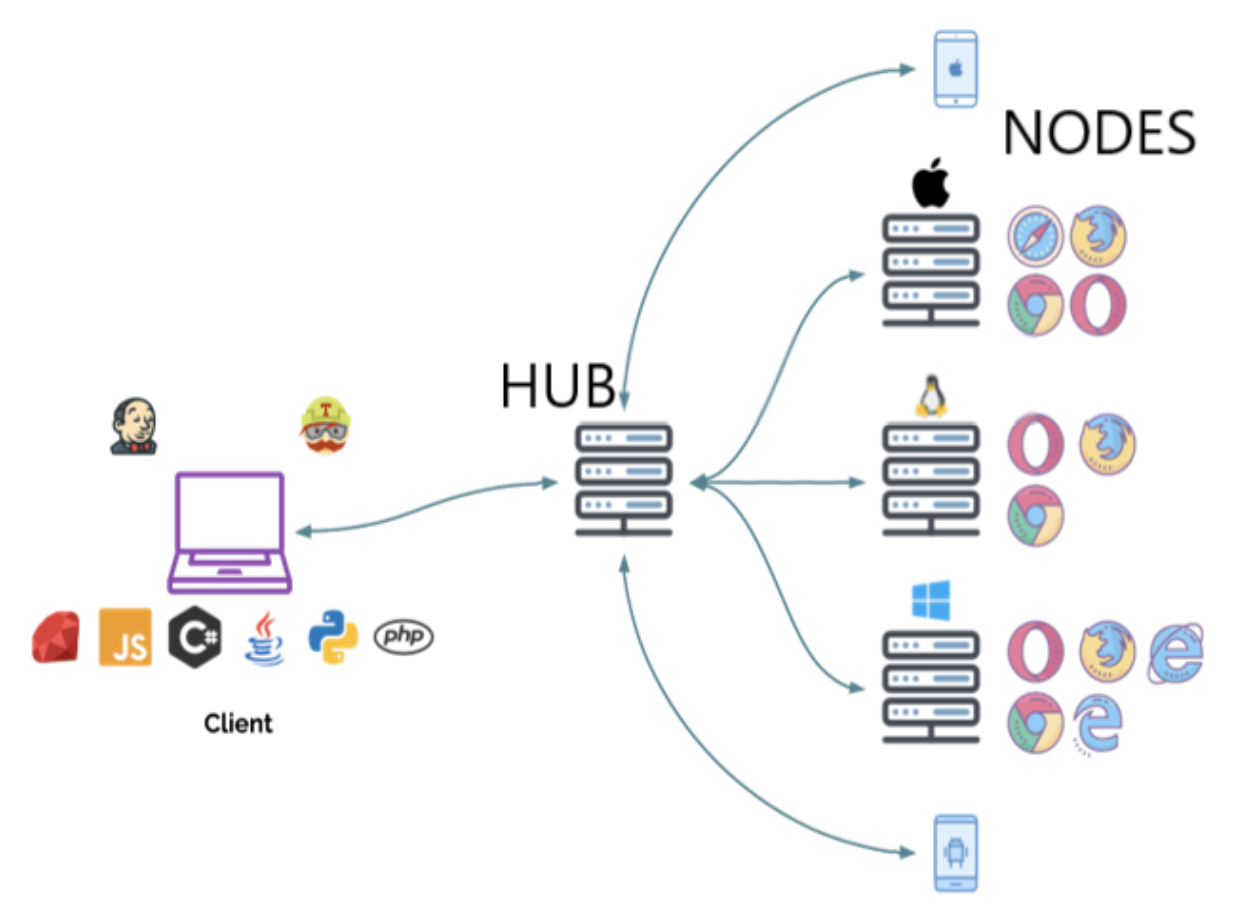
****

**What is Selenium Grid?**

The *Selenium Grid* is a testing tool which allows us to run our tests on different machines against different browsers. It is a part of the Selenium Suite which specialise in running multiple tests across different browsers, operating systems and machines. You can connect to it with Selenium Remote by specifying the browser, browser version, and operating system you want. You specify these values through *Selenium Remote’s Capabilities*.

The Selenium Grid is used because of many reasons. Here are a few

* When we want to run our tests against multiple browsers, the multiple versions of browsers and the browsers running on different operating systems.
* It is also used to reduce the time taken by the test suite to complete a test pass by running tests in parallel.

There are two main elements to Selenium Grid — a **hub**, and **nodes**.

**What is a Hub?**

In Selenium Grid, the *hub* is a computer which is the central point where we can load our tests into. Hub also acts as a server because of which it acts as a central point to control the network of Test machines. The Selenium Grid has only one hub and it is the master of the network. When a test with given Desired Capabilities is given to Hub, the Hub searches for the node which matches the given configuration.

For example, you can say that you want to run the test on Windows 10 and on Chrome browser with version XXX. Hub will try to find a machine in the Grid which matches the criterion and will run the test on that Machine. If there is no match, the hub returns an error. There should be only one hub in a Grid.

**What is a Node?**

In Selenium Grid, a *node* is referred to a Test Machine which opts to connect with the Hub. This test machine will be used by Hub to run tests on. A Grid network can have multiple nodes. A node is supposed to have different platforms i.e. different operating systems and browsers. The node does not need the same platform for running as that of the hub.

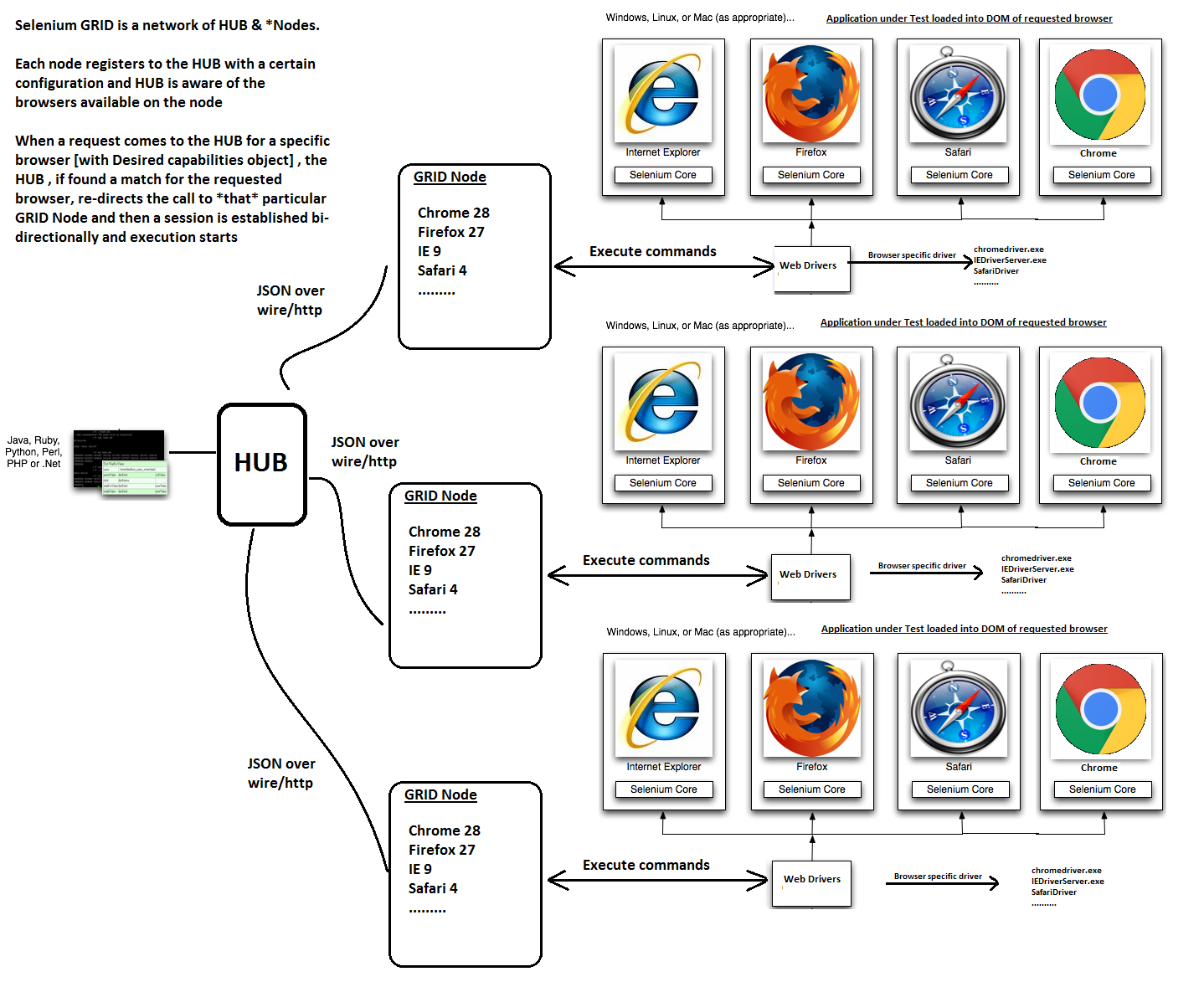
**How does it works?**

1. [Configure the **HUB**.](#oiehzl8hpjc5)
2. [Configure/connect the **NODES**.](#5xjzapippckf)
3. [Develop the **SCRIPT** and execute](#y9nzj5sqxcmk)

**Architecture and RemoteWebDriver WorkFlow**

RemoteWebDriver can be used the same way as WebDriver is used locally. The primary difference is that RemoteWebDriver needs to be configured so that it can run the tests on a separate machine. The RemoteWebDriver is composed of two pieces: a client and a server. The client is the WebDriver test and the server is simply a Java servlet, which can be hosted in any modern JEE app server.

* RemoteWebDriver is an implementation class of the WebDriver interface that is used to execute test scripts on remote machines.
* There are two parts to RemoteWebDriver: a server(hub) and a client(node)
  + The RemoteWebDriverserver is a component that listens on a port for various requests from a RemoteWebDriver Once it receives the requests, it forwards them to any of the following: Firefox Driver, IE Driver, or Chrome Driver, whichever is asked.
  + The language-binding client libraries that serve as a RemoteWebDriver The client, as it used to when executing tests locally, translates your test script requests to JSON payload and sends them across to the RemoteWebDriverserver using the JSON wire protocol.
* When you execute your tests locally, the WebDriver client libraries talk to your Firefox Driver, IE Driver, or Chrome Driver directly. Now, when you try to execute your tests remotely, the WebDriver client libraries talk to the RemoteWebDriverserver and the server talks to either the Firefox Driver, IE Driver, or Chrome Driver, whichever the WebDriver client asks for.



**SETUP - SELENIUM GRID - HUB & NODES**

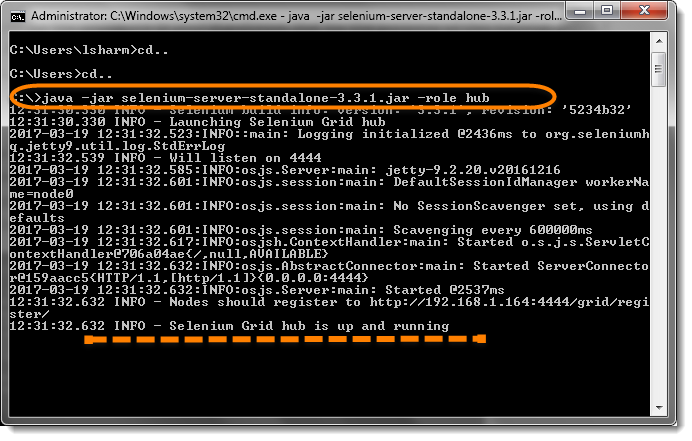
**Step 1 : Download Selenium Server and Set up Selenium Grid Hub**

1. Download the latest Selenium Server file from → [**Downloads**](http://docs.seleniumhq.org/download/)

***\*Prerequisite : JAVA is required to run SELENIUM SERVER***

1. Place the **Selenium Server JAR file** anywhere in your FOLDER STRUCTURE.
2. LAUNCH the JAR file from Command Prompt by navigating to the folder where we placed and RUN command:

**$ java -jar selenium-server-standalone-3.3.1.jar -role hub**

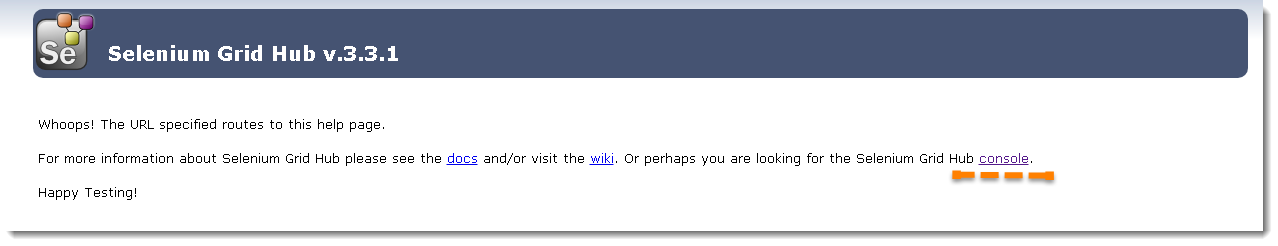
****

**Note:**

* Make sure you change the **version** number from the command accordingly.
* Closing the command prompt window will also stop the selenium server
* Selenium Grid, default port **4444** for its web interface. To change port, use this command:

**$ java -jar selenium-server-standalone-3.3.1.jar -port 4455 -role hub**

1. Verify HUB is running, open [**http://localhost:4444**](http://localhost:4444). You should see something like below:

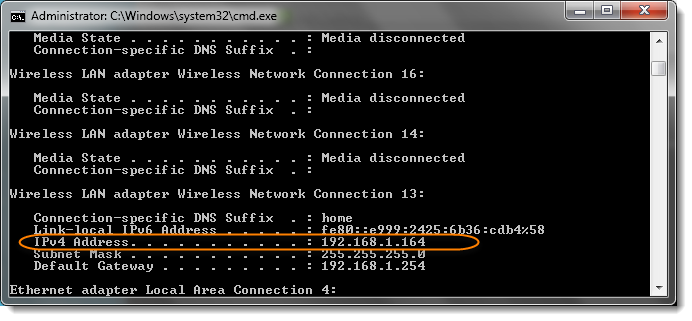


**Console gives the information about what is available on the Hub. As of now it will be blank, as there is no machine connected to it.**

**Step 2 : Set up Node Machine**

Ideally the Node Machine has to be different from the Hub Machine, but just to maintain the simplicity, we can set up Node on the same machine where the Hub is running. Steps are completely the same, except that the IP address changes to the Node Machine’s IP Address. It is required to download the Selenium Server JAR file on the Node Machine.

1. IP Address of the Hub Machine. Go to Command Prompt and type IPCONFIG to find out the IP Address.

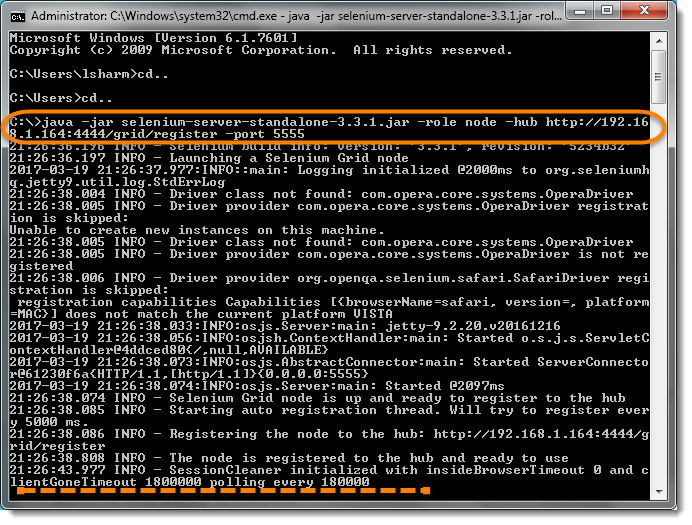


This means that the IP Address of the Hub Machine will become:

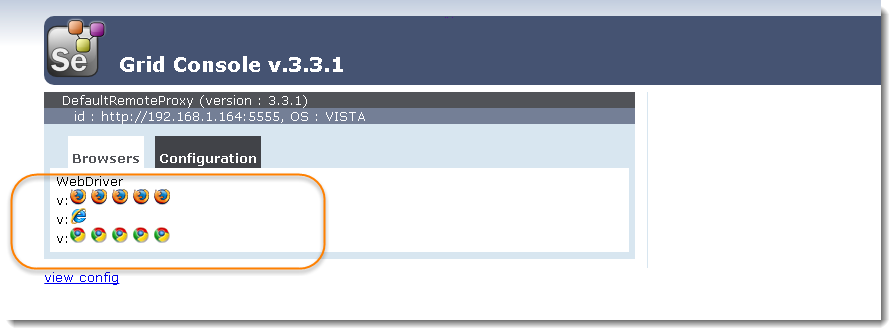
**http:// + Hub Machine IP Address + Hub Port = http://192.168.1.164:4444**

1. If you are setting up Node on a different machine, log on to that machine and open Command Prompt and register Hub Machine with Node Machine;

$ **java -jar selenium-server-standalone-3.3.1.jar -role node -hub http://192.168.1.164:4444/grid/register -port 5555**

****

1. After executing the command then return to the Hub and navigate the URL **http://localhost:4444 or http://192.168.1.164:4444** and the hub will now display the node which is attached to it.



**Note :** The above Console page gives the information about the Node Machines, which all are connected to the Hub. It provides the information about the Node Machine IP Address, OS Type, Browsers etc. You will find 5 Chrome, 5 Firefox and 1 IE browser under the Browser section like above. This indicates that by default you can use 5 Chrome, 5 Firefox and 1 IE browser. In case of more machines attached, you would see more blocks on the Console Page.

**Step 3 : Write a Test Script**

Sample WebDriver Code that can be run on the HUB and automation code will be executed on the NODE machine.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | import org.openqa.selenium.WebDriver;  import org.openqa.selenium.remote.DesiredCapabilities;  import org.openqa.selenium.remote.RemoteWebDriver;    import java.net.MalformedURLException;  import java.net.URL;    public class Grid\_SetUp  {  public static WebDriver driver;    public static void main(String[] args) throws MalformedURLException, InterruptedException  {  String URL = "http://www.DemoQA.com";  String Node = "http://192.168.1.164:4444/wd/hub";  **DesiredCapabilities cap = DesiredCapabilities.firefox();**    **driver = new RemoteWebDriver(new URL(Node), cap);**    driver.navigate().to(URL);  Thread.sleep(5000);  driver.quit();  }  } |