Execute webdriver tests in Parallel using selenium Grid

Home >> Selenium Tutorials >> Execute webdriver tests in Parallel using selenium Grid

In previous article we have seen configuring selenium grid and execute a simple test on firefox browser. In this tutorial we will see 'Parallel execution of tests' using selenium grid and execute tests on firefox and chrome browser.

We will register multiple nodes to the Hub and execute tests in parallel. In the below example we will register a node in the same local machine where hub is running and other node in remote machine.

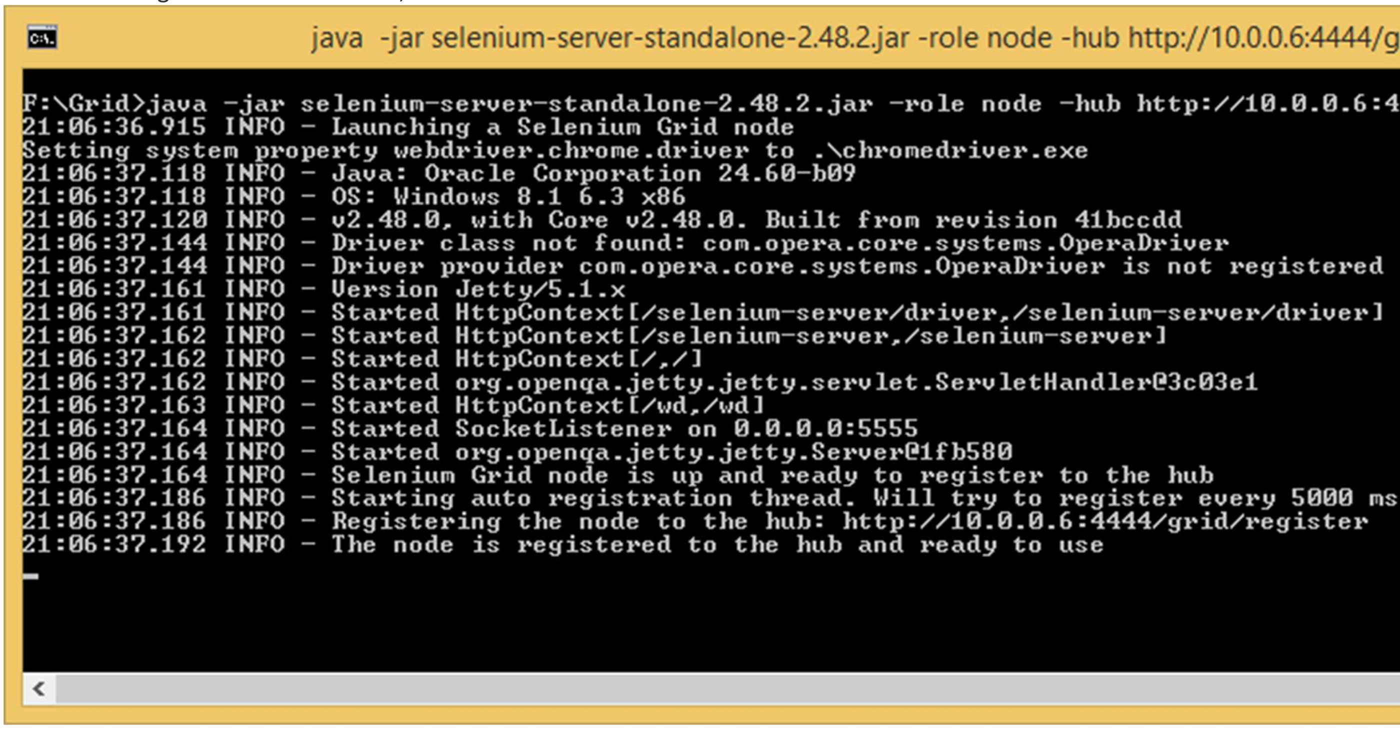
We will register ***node 1*** in local machine with the below command:

java -jar selenium-server-standalone-2.48.2.jar -role node -hub http://localhost:4444/grid/register

We will register ***node 2*** with the below command in remote machine. In the below command, need to pass a parameter by passing JVM properties using the -D flag along with chrome driver path , so that when ever there is a request to execute in chrome driver, Hub will send the request to this node. And here we have to mention the IP address of the machine where the hub is running as we are starting this node in remote machine.

java -jar selenium-server-standalone-2.48.2.jar -role node -hub http://10.0.0.6:4444/grid/register -Dwebdriver.chrome.driver=.\chromedriver.exe

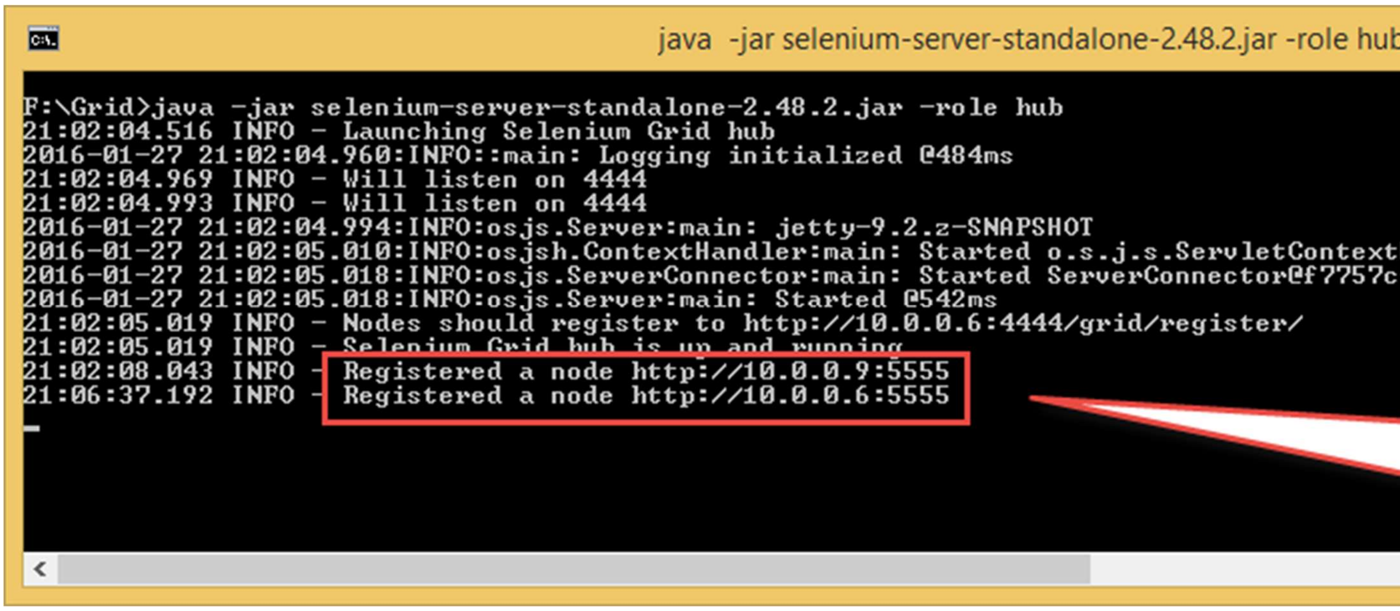
After executing the above command, console should look like below:



```
java  -jar selenium-server-standalone-2.48.2.jar -role node -hub http://10.0.0.6:4444/g

F:\Grid>java -jar selenium-server-standalone-2.48.2.jar -role node -hub http://10.0.0.6:4
21:06:36.915 INFO - Launching a Selenium Grid node
Setting system property webdriver.chrome.driver to .\chromedriver.exe
21:06:37.118 INFO - Java: Oracle Corporation 24.60-b09
21:06:37.118 INFO - OS: Windows 8.1 6.3 x86
21:06:37.120 INFO - v2.48.0, with Core v2.48.0. Built from revision 41bccdd
21:06:37.144 INFO - Driver class not found: com.opera.core.systems.OperaDriver
21:06:37.144 INFO - Driver provider com.opera.core.systems.OperaDriver is not registered
21:06:37.161 INFO - Version Jetty/5.1.x
21:06:37.161 INFO - Started HttpContext[/selenium-server/driver,/selenium-server/driver]
21:06:37.162 INFO - Started HttpContext[/selenium-server,/selenium-server]
21:06:37.162 INFO - Started HttpContext[/,/]
21:06:37.162 INFO - Started org.openqa.jetty.servlet.ServletHandler@3c03e1
21:06:37.163 INFO - Started HttpContext[/wd,/wd]
21:06:37.164 INFO - Started SocketListener on 0.0.0.0:5555
21:06:37.164 INFO - Started org.openqa.jetty.jetty.Server@1fb580
21:06:37.164 INFO - Selenium Grid node is up and ready to register to the hub
21:06:37.186 INFO - Starting auto registration thread. Will try to register every 5000 ms
21:06:37.186 INFO - Registering the node to the hub: http://10.0.0.6:4444/grid/register
21:06:37.192 INFO - The node is registered to the hub and ready to use
_
```

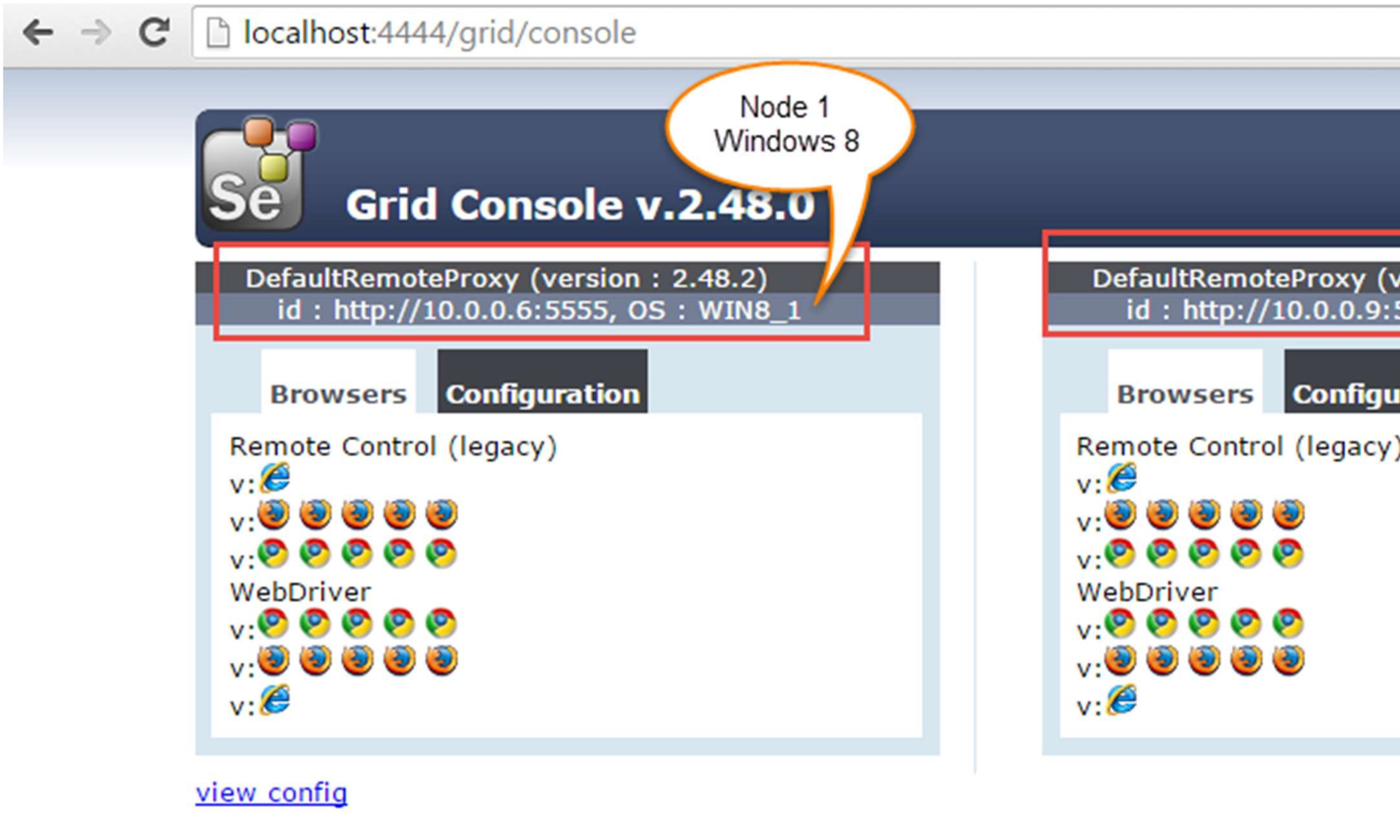
When a Hub receives request to execute test in Chrome browser and If we don't specify chrome driver path for node, it will throw an exception as "Exception: The path to the chromedriver executable must be set by the webdriver.chrome.driver system property; for more information, see http://code.google.com/p/selenium/wiki/ChromeDriver. The latest version can be downloaded from http://code.google.com/p/chromedriver/downloads/list".

Now when these two nodes are registered to the hub, hub console should look like below :



```
java  -jar selenium-server-standalone-2.48.2.jar -role hub
F:\Grid>java -jar selenium-server-standalone-2.48.2.jar -role hub
21:02:04.516 INFO - Launching Selenium Grid hub
2016-01-27 21:02:04.960:INFO::main: Logging initialized @484ms
21:02:04.969 INFO - Will listen on 4444
21:02:04.993 INFO - Will listen on 4444
2016-01-27 21:02:04.994:INFO:osjs.Server:main: jetty-9.2.z-SNAPSHOT
2016-01-27 21:02:05.010:INFO:osjsh.ContextHandler:main: Started o.s.j.s.ServletContext
2016-01-27 21:02:05.018:INFO:osjs.ServerConnector:main: Started ServerConnector@f7757c
2016-01-27 21:02:05.018:INFO:osjs.Server:main: Started @542ms
21:02:05.019 INFO - Nodes should register to http://10.0.0.6:4444/grid/register/
21:02:05.019 INFO - Selenium Grid hub is up and running
21:02:08.043 INFO - Registered a node http://10.0.0.9:5555
21:06:37.192 INFO - Registered a node http://10.0.0.6:5555
```

And the Grid console should look like below which shows configuration details of the nodes with IP address. In the below image below image Node 1 is running in Windows * machine (WIN8_1) and Node 2 is running in Windows7 machine.



Let us now take the example and execute tests in parallel.
In order to do this, we will create two classes which has multiple @Test methods in it. And a Browser class which invokes the remote webdriver based on the browser parameter that we pass in testng.xml file
First create a class as 'Browser.java'

```
package com.test;

import java.net.MalformedURLException;
import java.net.URL;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.remote.RemoteWebDriver;

public class Browser {

        public static RemoteWebDriver getDriver(String browser) throws MalformedURLException {
                return new RemoteWebDriver(new URL("http://10.0.0.6:4444/wd/hub"), getBrowserCa
pabilities(browser));
        }

        private static DesiredCapabilities getBrowserCapabilities(String browserType) {
                switch (browserType) {
                case "firefox":
                        System.out.println("Opening firefox driver");
                        return DesiredCapabilities.firefox();
                case "chrome":
                        System.out.println("Opening chrome driver");
                        return DesiredCapabilities.chrome();
                case "IE":
                        System.out.println("Opening IE driver");
                        return DesiredCapabilities.internetExplorer();
                default:
                        System.out.println("browser : " + browserType + " is invalid, Launching Firefo
x as browser of choice..");
                        return DesiredCapabilities.firefox();
                }
        }
}
```

We will call 'getDriver' method which will intern call 'getBrowserCapabilities' based on browser parameter from the below two classes. If we pass 'chrome' as parameter, it is invoke chromedriver
Let us create a class as 'ParallelTestA.java' as below:

```
package com.test;

import java.net.MalformedURLException;

import org.openqa.selenium.remote.RemoteWebDriver;
import org.testng.Assert;
import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Parameters;
import org.testng.annotations.Test;

public class ParallelTestA {

        public static RemoteWebDriver driver;
        public static String appURL = "http://www.google.com";

        @BeforeClass
        @Parameters({ "browser" })
        public void setUp(String browser) throws MalformedURLException {
                System.out.println("*******************");
                driver = Browser.getDriver(browser);
                driver.manage().window().maximize();
        }

        @Test
```

```
	public void testGooglePageTitleInFirefox() {
		driver.navigate().to(appURL);
		String strPageTitle = driver.getTitle();
		Assert.assertTrue(strPageTitle.equalsIgnoreCase("Google"), "Page title doesn't match");
	}

	@AfterClass
	public void tearDown() {
		if(driver!=null) {
			System.out.println("Closing browser");
			driver.quit();
		}
	}

}
```

Let us create a class as 'ParallelTestB.java' as below:

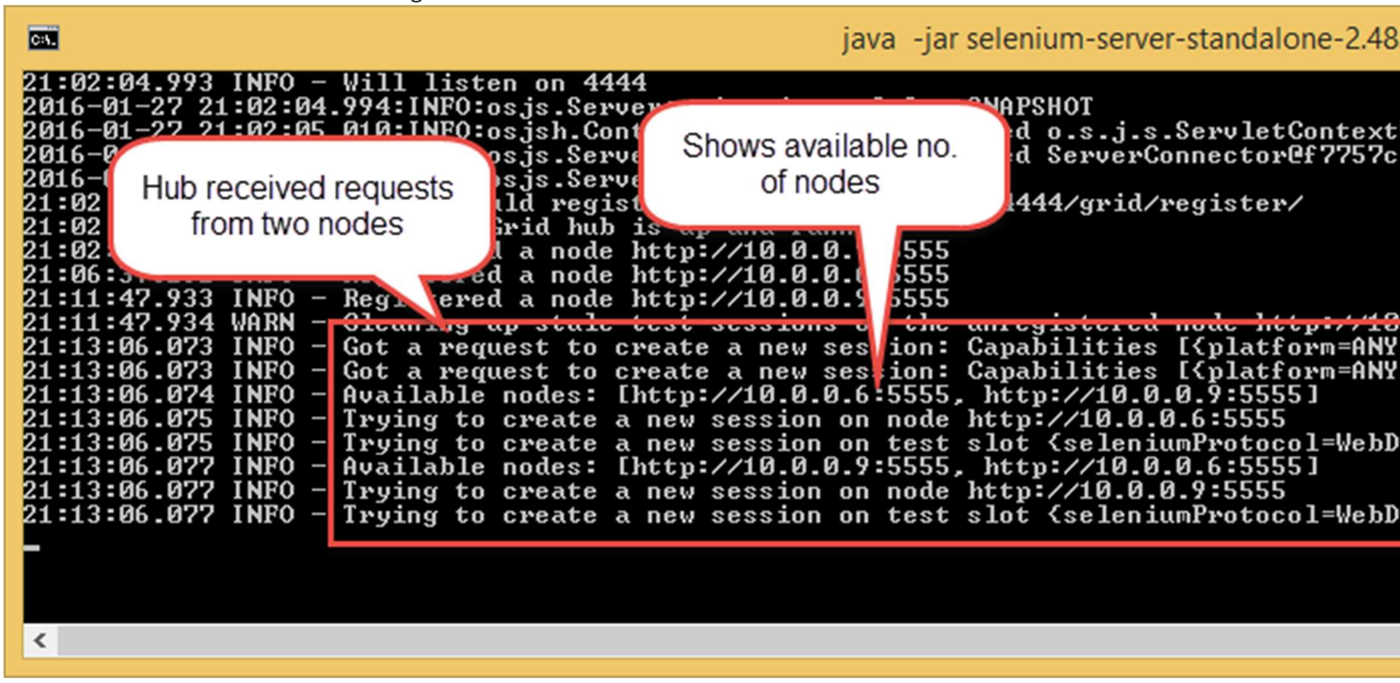
```
package com.test;

import java.net.MalformedURLException;

import org.openqa.selenium.By;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.testng.Assert;
import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Parameters;
import org.testng.annotations.Test;

public class ParallelTestB {

	public static RemoteWebDriver driver;
	public static String appURL = "http://www.google.com";

	@BeforeClass
	@Parameters({ "browser" })
	public void setUp(String browser) throws MalformedURLException {
		System.out.println("*******************");
		driver = Browser.getDriver(browser);
		driver.manage().window().maximize();
	}

	@Test
	public void testGooglePageTitleInChrome() {
		driver.navigate().to("http://www.google.com");
		String strPageTitle = driver.getTitle();
		Assert.assertTrue(strPageTitle.equalsIgnoreCase("Google"), "Page title doesn't match");
	}

	@Test
	public void testSearchGoogle() {
		System.out.println("Opening Google..");
		driver.navigate().to(appURL);
		driver.findElement(By.name("q")).sendKeys("Selenium Easy Grid Tutorials");
		driver.findElement(By.name("btnG")).click();
	}
```

```java
		@AfterClass
		public void tearDown() {
			if(driver!=null) {
				System.out.println("Closing browser");
				driver.quit();
			}
		}
}
```

Now to execute these tests, we need to create testng.xml file as below and set parallel="tests" with parameter browser for each test

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >
<suite name="Main Test Suite" parallel="tests" verbose="1">
  <test name="Grid firefox Test">
  <parameter name="browser" value="firefox"/>
    <classes>
    <class name="com.test.ParallelTestA"/>
    </classes>
  </test>
  <test name="Grid chrome Test">
  <parameter name="browser" value="chrome"/>
    <classes>
    <class name="com.test.ParallelTestB"/>
    </classes>
  </test>
</suite>
```

Once we execute the above code, hub console will display information such as on 'number of nodes available', on which node it has started executing the tests etc.. as below :

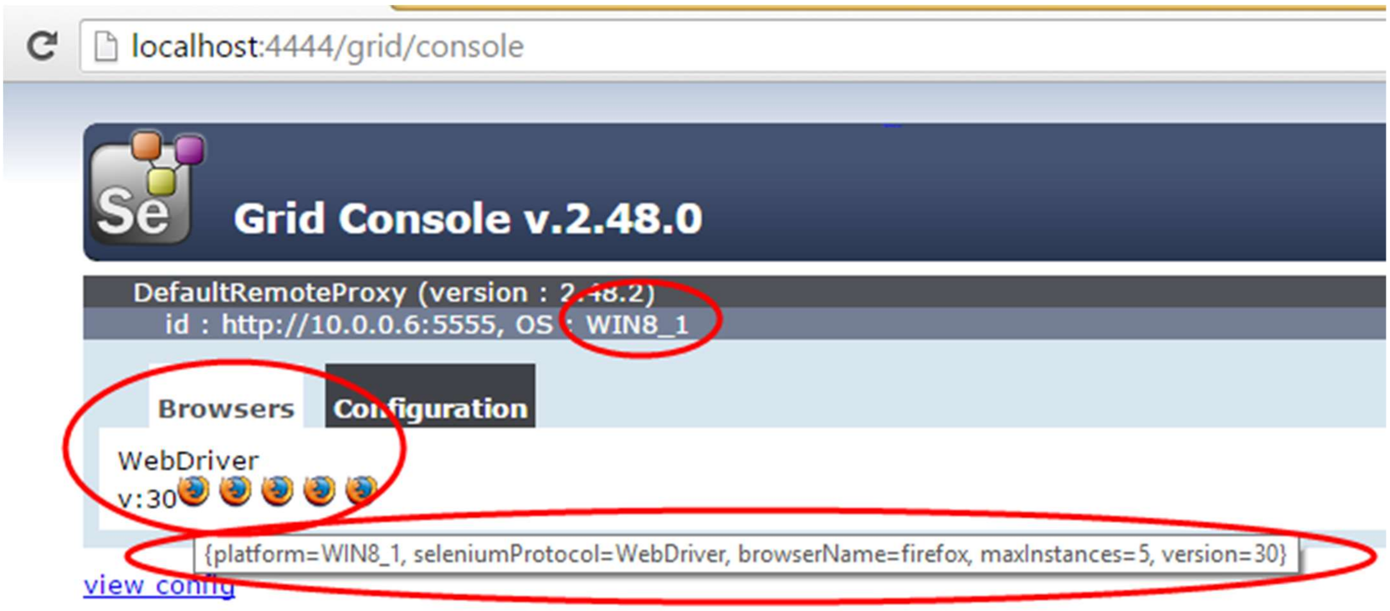

java -jar selenium-server-standalone-2.48
Hub received requests from two nodes
Shows available no. of nodes

```
21:02:04.993 INFO - Will listen on 4444
2016-01-27 21:02:04.994:INFO:osjs.Server                        SNAPSHOT
2016-01-27 21:02:05.010:INFO:osjsh.Cont                         d o.s.j.s.ServletContext
2016-0                     osjs.Serve                          d ServerConnector@f7757c
2016-                      osjs.Serve
21:02                      uld regist                          444/grid/register/
21:02                      Grid hub is
21:02                      d a node http://10.0.0.    555
21:06:                     ered a node http://10.0.0.  5555
21:11:47.933 INFO - Reg    ered a node http://10.0.0.  5555
21:11:47.934 WARN -
21:13:06.073 INFO - Got a request to create a new ses ion: Capabilities [{platform=ANY
21:13:06.073 INFO - Got a request to create a new ses ion: Capabilities [{platform=ANY
21:13:06.074 INFO - Available nodes: [http://10.0.0.6:5555, http://10.0.0.9:5555]
21:13:06.075 INFO - Trying to create a new session on node http://10.0.0.6:5555
21:13:06.075 INFO - Trying to create a new session on test slot {seleniumProtocol=WebD
21:13:06.077 INFO - Available nodes: [http://10.0.0.9:5555, http://10.0.0.6:5555]
21:13:06.077 INFO - Trying to create a new session on node http://10.0.0.9:5555
21:13:06.077 INFO - Trying to create a new session on test slot {seleniumProtocol=WebD
_
```

There are many other parameters for browser settings that we can pass when registering node to hub. When ever we use **-browser** parameter, the default browsers will be ignored and only what we specify in command line will be used.
Example :

```
-browser browserName=firefox,version=30,maxInstances=5,platform=WIN8_1
```

After executing the above command, if you check the grid console, it will just show the browserName, browser version, Max instances (number of instances of same version of browser you can run over the Remote System) and the platform (OS running on the node) that we have used to register the node. Grid console looks like below :



DefaultRemoteProxy (version : 2.48.2)
id : http://10.0.0.6:5555, OS : WIN8_1

WebDriver
v:30

{platform=WIN8_1, seleniumProtocol=WebDriver, browserName=firefox, maxInstances=5, version=30}

If the remote machine has multiple versions of Firefox, We can map the location of each binary to a particular version on the same machine to execute with multiple versions as below:

```
-browser browserName=firefox,version=40,firefox_binary=c:\Program Files\firefox30\firefox,maxInstances=3,
platform=WINDOWS -browser browserName=firefox,version=41,firefox_binary=c:\Program Files\firefox40\fire
fox,maxInstances=3,platform=WINDOWS
```