**Selenium GRID SetUp Detailed steps**

* You would have to make sure that your system has Java Runtime Environment (JRE) or better yet Java Development Kit (JDK) installed. Though most people recommend going for the latest JDK, I prefer an earlier tried and tested versions like JDK SE 08 or 09. You can go for the latest one if you wish.
* Download and extract Selenium Standalone server JAR files. You can download them from [here](https://www.seleniumhq.org/download/).

Configuring A Selenium Hub

The first step of a **Selenium Grid** setup would be to create a hub. Open a command prompt or terminal and navigate to the directory where the Selenium Standalone Server jar file is saved. Run the below command.

|  |  |
| --- | --- |
| 1 | java -jar selenium-server-standalone-3.141.59.jar -role hub |

This command would launch a Selenium Grid hub on port 4444 by default. You can also check the same by directing to *http://localhost:4444/grid/console*.

Configuring Nodes For Selenium Grid

Since the Hub has already been created, the next step to **Selenium Grid** setup for parallel execution would involve launching a node. You start with going to the other machines where we want to setup nodes. Again, these machines should have a JDK/JRE setup.

In the node machine, open a command prompt or terminal and navigate to the directory where you have saved the browser driver files.

Configuring ChromeDriver In Selenium Grid Setup

Enter the below command in order to configure Chrome driver in your Selenium Grid setup for parallel execution.

|  |  |
| --- | --- |
| 1  2  3 | java -Dwebdriver.chrome.driver="C:\chromedriver.exe" -jar  selenium-server-standalone-3.141.59.jar -role webdriver -hub  http://10.0.0.22:4444/grid/register -port 4546 |

**Note**: In the above command I have used my machine IP along with the port 4444 where the hub is actually running. I have also selected the port 4546 where I want my node to get registered. You can opt for any free port for registering the node.

Similarly, you can register other nodes for other browsers. For example, here’s how you can configure Firefox(Gecko) driver and Internet Explorer driver to your **Selenium Grid setup** for parallel execution.

Configuring Firefox(Gecko) Driver In Selenium Grid Setup

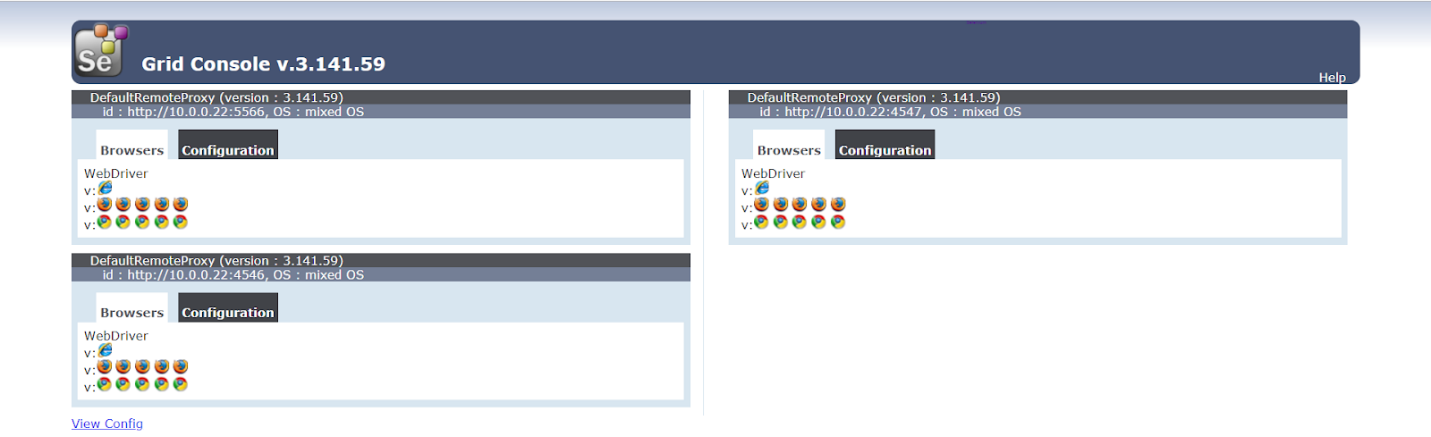
|  |  |
| --- | --- |
| 1  2  3 | java -Dwebdriver.gecko.driver="C:\geckodriver.exe" -jar  selenium-server-standalone-3.141.59.jar -role webdriver -hub  http://10.0.0.22:4444/grid/register -port 5566 |

Configuring IE(Internet Explorer) Driver In Selenium Grid Setup

|  |  |
| --- | --- |
| 1  2  3 | java -Dwebdriver.ie.driver="C:\IEDriverServer.exe" -jar  selenium-server-standalone-3.141.59.jar -role webdriver -hub  http://10.0.0.22:4444/grid/register -port 4547 |

**Note:** Please make sure that you select different ports for different nodes to connect with the single hub running on port 4444. Also, before running your test on IE, make sure that the browser zoom is exactly 100% otherwise it might show an error.

You can notice by above commands that we are running our first node for chrome driver on port 4546, the second node for firefox driver is running on port 5566, and the third node for IE browser is running on port 4547. Here is the output screenshot for further reference:



Now, that we have our Selenium Grid setup configured, your next step is to perform Selenium automation testing for your cross browser testing suite.

**Run Selenium Grid For Parallel Automation Testing**

It is time for some automated cross browser testing! I will be running an automation script demonstrating the Selenium testing Grid for parallel execution. This script would run in parallel on Chrome, Firefox, and Internet Explorer which are registered on different ports and attached to a single hub.

Here is the sample code I used to perform [automation testing](https://www.lambdatest.com/automation-testing) with **Selenium Grid** for parallel execution in different browsers.

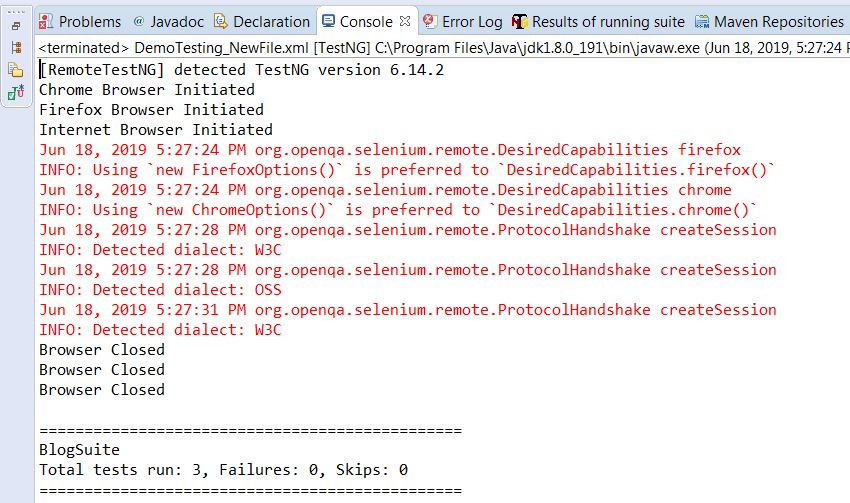
|  |  |
| --- | --- |
|  | package DemoAutomation; |
|  |  |
|  | import java.net.MalformedURLException; |
|  | import java.net.URL; |
|  | import java.util.concurrent.TimeUnit; |
|  |  |
|  | import org.openqa.selenium.By; |
|  | import org.openqa.selenium.Platform; |
|  | import org.openqa.selenium.WebDriver; |
|  | import org.openqa.selenium.ie.InternetExplorerDriver; |
|  | import org.openqa.selenium.remote.CapabilityType; |
|  | import org.openqa.selenium.remote.DesiredCapabilities; |
|  | import org.openqa.selenium.remote.RemoteWebDriver; |
|  | import org.testng.annotations.AfterMethod; |
|  | import org.testng.annotations.BeforeMethod; |
|  | import org.testng.annotations.DataProvider; |
|  | import org.testng.annotations.Parameters; |
|  | import org.testng.annotations.Test; |
|  |  |
|  |  |
|  | public class Selenium\_MultiBrowser\_Test |
|  | { |
|  | WebDriver driver; |
|  | String nodeURL; |
|  |  |
|  | @Parameters({"Port"}) |
|  | @BeforeMethod() |
|  | public void setUp(String Port) throws MalformedURLException |
|  | { |
|  | if(Port.equalsIgnoreCase("4546")) |
|  | { |
|  | nodeURL = "http://10.0.0.22:4546/wd/hub"; |
|  | System.out.println("Chrome Browser Initiated"); |
|  | DesiredCapabilities capabilities = DesiredCapabilities.chrome(); |
|  | capabilities.setBrowserName("chrome"); |
|  | capabilities.setPlatform(Platform.WINDOWS); |
|  |  |
|  | driver = new RemoteWebDriver(new URL(nodeURL),capabilities); |
|  |  |
|  | driver.get("https://www.apple.com/"); |
|  | driver.manage().window().maximize(); |
|  | driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS); |
|  | } |
|  |  |
|  | else |
|  | if(Port.equalsIgnoreCase("5566")) |
|  | { |
|  | nodeURL = "http://10.0.0.22:5566/wd/hub"; |
|  | System.out.println("Firefox Browser Initiated"); |
|  | DesiredCapabilities capabilities1 = DesiredCapabilities.firefox(); |
|  | capabilities1.setBrowserName("firefox"); |
|  | capabilities1.setPlatform(Platform.WINDOWS); |
|  |  |
|  | driver = new RemoteWebDriver(new URL(nodeURL),capabilities1); |
|  |  |
|  | driver.get("https://www.apple.com/"); |
|  | driver.manage().window().maximize(); |
|  | driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS); |
|  | } |
|  |  |
|  | else |
|  |  |
|  | if(Port.equalsIgnoreCase("4547")) |
|  | { |
|  | nodeURL = "http://10.0.0.22:4547/wd/hub"; |
|  | System.out.println("Internet Browser Initiated"); |
|  | DesiredCapabilities capabilities2 = DesiredCapabilities.internetExplorer(); |
|  | capabilities2.setBrowserName("internet explorer"); |
|  | capabilities2.setCapability(InternetExplorerDriver.INTRODUCE\_FLAKINESS\_BY\_IGNORING\_SECURITY\_DOMAINS, true); |
|  | capabilities2.setCapability(InternetExplorerDriver.IGNORE\_ZOOM\_SETTING, true); |
|  | capabilities2.setCapability(CapabilityType.ACCEPT\_SSL\_CERTS, true); |
|  | capabilities2.setCapability("ignoreProtectedModeSettings", true); |
|  | capabilities2.setCapability("nativeEvents", false); |
|  | capabilities2.setCapability(InternetExplorerDriver.INITIAL\_BROWSER\_URL, ""); |
|  | capabilities2.setCapability(InternetExplorerDriver.LOG\_LEVEL, "DEBUG"); |
|  |  |
|  |  |
|  | capabilities2.setPlatform(Platform.WINDOWS); |
|  |  |
|  | driver = new RemoteWebDriver(new URL(nodeURL),capabilities2); |
|  |  |
|  | driver.get("https://www.apple.com/"); |
|  | driver.manage().window().maximize(); |
|  | driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS); |
|  | } |
|  | } |
|  |  |
|  | @Test |
|  | public void appleSite() throws InterruptedException |
|  | { |
|  | try |
|  | { |
|  |  |
|  | driver.findElement(By.xpath("//\*[@id=\'ac-globalnav\']/div/ul[2]/li[3]")).click(); |
|  | Thread.sleep(2000); |
|  |  |
|  | driver.findElement(By.cssSelector("#chapternav > div > ul > li.chapternav-item.chapternav-item-ipad-air > a > figure")).click(); |
|  | Thread.sleep(2000); |
|  |  |
|  | driver.findElement(By.linkText("Why iPad")).click(); |
|  | Thread.sleep(2000); |
|  | } |
|  |  |
|  | catch(Exception e) |
|  | { |
|  | System.out.println(e.getMessage()); |
|  | } |
|  | } |
|  |  |
|  |  |
|  | @AfterMethod() |
|  | public void tearDown() |
|  | { |
|  | driver.quit(); |
|  | System.out.println("Browser Closed"); |
|  | } |
|  | } |

Above is the java class file that is configured with the XML file which includes the values of parameters passed in the java file and also helps in creating a suite of different classes that would run in a parallel manner.

|  |  |
| --- | --- |
|  | <?xml version="1.0" encoding="UTF-8"?> |
|  | <!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd"> |
|  | <suite thread-count="3" name="BlogSuite" parallel="tests"> |
|  |  |
|  | <test name="Chrome Test"> |
|  | <parameter name="Port" value="4546"/> |
|  | <classes> |
|  | <class name="DemoAutomation.Selenium\_MultiBrowser\_Test"/> |
|  |  |
|  | </classes> |
|  | </test> |
|  |  |
|  | <test name="Firefox Test"> |
|  | <parameter name="Port" value="5566"/> |
|  | <classes> |
|  | <class name="DemoAutomation.Selenium\_MultiBrowser\_Test"/> |
|  |  |
|  | </classes> |
|  | </test> |
|  |  |
|  |  |
|  | <test name="Internet Explorer Test"> |
|  | <parameter name="Port" value="4547"/> |
|  | <classes> |
|  | <class name="DemoAutomation.Selenium\_MultiBrowser\_Test"/> |
|  |  |
|  | </classes> |
|  | </test> |
|  |  |
|  |  |
|  | </suite> |

[view raw](https://gist.github.com/muditlambda/b737489997186337651b8f7fdda30a0c/raw/c2cdd2ba1ef50e2673dba75b89f7c086726d0694/parallel.xml)[parallel.xml](https://gist.github.com/muditlambda/b737489997186337651b8f7fdda30a0c#file-parallel-xml)hosted with ❤ by [GitHub](https://github.com/)

**Output Screen:**



In the above code, I have used **DesiredCapabilities** class that would help you set properties for the Selenium WebDriver. These properties can be used to configure instances of browsers such as BrowserName and BrowserVersion on which you want your script to run.