Selenium Grid for Parallel Execution

When we say parallel test execution in Selenium is achieved by Selenium Grid than statement is partly incorrect.

Testing Framework like testng is used for parallel test execution

Selenium Grid is used for automated testing execution on Distributed systems parallely

Selenium Grid Concept

In Selenium Grid architecture we have 1 Hub which acts as central controlling authority and connecting nodes. Nodes must be registered to Hub

Consider node as port opened on machine (loacal or remote). Each node is capable to opening multiple browsers.

On single machine we can have multiple nodes opened

Grid Hub decides what tests needs to routed on which node, we can’t control them

Best Practices of Selenium Grid

Single machine should open one node only

Each node should run only single type of browser

We need various driver objects for various threads to be run parallely, so create driver as ThreadLocal variable

Let consider an example what we would be achieving in our Grid:

SeleniumGrid

Start Hub and Nodes

Download “Selenium Standalone Server” from “http://www.seleniumhq.org/download/” on all 3 machines

Goto machine 1 and open command prompt.

Navigate to location where jar is located.

Start Hub by command: java -jar selenium-server-standalone-2.48.2.jar -role hub

Open browser and navigate to http://localhost:4444/grid/console and verify hub is started by checking below image.HubStarted

Goto machine 2 and open command prompt

Navigate to location where jar is located.

Start Node by command: java -Dwebdriver.chrome.driver={path to chromedriver.exe}-jar selenium-server-standalone-2.48.2.jar -role webdriver -hub -port 5560 -browser browserName=chrome,maxInstances=2,maxSession=2

Goto machine 1 and in browser navigate to http://localhost:4444/grid/console and verify node is started by checking below image.SeleniumGridNode1

Goto machine 3 and open command prompt.

Navigate to location where jar is located.

Start Node by command: java -jar selenium-server-standalone-2.48.2.jar -role webdriver -hub -port 5557 -browser browserName=firefox,maxInstances=5,maxSession=2

Goto machine 1 and in browser navigate to http://localhost:4444/grid/console and verify node is started by checking below image. SeleniumGridNode2

Nodes can be opened for various settings like browser, platform, version.

Now Selenium Hub and Nodes are created, lets make some @Test and execute them.

Make Project on any machine, my project structure as below:PackageStructure

Source code for files are given below.

Execute testng.xml and execution will start parallely on both machines.

testng.xml

<?xml version="1.0" encoding="UTF-8"?>

<suite name="Parallel test suite" parallel="classes" thread-count="2">

<test name="Regression 1">

<parameter name="myBrowser" value="firefox"/>

<classes>

<class name="myPackage.TestParallel" />

<class name="myPackage.TestParallel" />

</classes>

</test>

<test name="Regression 2">

<parameter name="myBrowser" value="chrome"/>

<classes>

<class name="myPackage.TestParallel" />

<class name="myPackage.TestParallel" />

</classes>

</test>

</suite>

**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"), getBrowserCapabilities(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 Firefox as browser of choice..");

return DesiredCapabilities.firefox();

}

}

}

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

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();

}

@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 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>