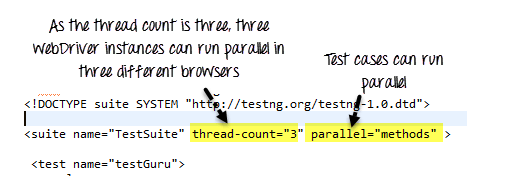
## **How to run Parallel Tests with Selenium**

There are situations where you want to run multiple tests at the same time.In such cases, one can use “parallel” attribute



The parallel attribute of suite tag can accept four values:

|  |  |
| --- | --- |
| tests | All the test cases inside <test> tag of[Testing](https://www.guru99.com/software-testing.html)xml file will run parallel. |
| classes | All the test cases inside a[Java](https://www.guru99.com/java-tutorial.html)class will run parallel |
| methods | All the methods with @Test annotation will execute parallel. |
| instances | Test cases in same instance will execute parallel but two methods of two different instances will run in different thread. |

The attribute *thread-count* allows you to specify how many threads should be allocated for this execution.

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.annotations.Test;

public class TestGuru99MultipleSession {

@Test

public void executSessionOne(){

//First session of WebDriver

System.setProperty("webdriver.chrome.driver","chromedriver.exe");

WebDriver driver = new ChromeDriver();

//Goto guru99 site

driver.get("http://demo.guru99.com/V4/");

//find user name text box and fill it

driver.findElement(By.name("uid")).sendKeys("Driver 1");

}

@Test

public void executeSessionTwo(){

//Second session of WebDriver

System.setProperty("webdriver.chrome.driver","chromedriver.exe");

WebDriver driver = new ChromeDriver();

//Goto guru99 site

driver.get("http://demo.guru99.com/V4/");

//find user name text box and fill it

driver.findElement(By.name("uid")).sendKeys("Driver 2");

}

@Test

public void executSessionThree(){

//Third session of WebDriver

System.setProperty("webdriver.chrome.driver","chromedriver.exe");

WebDriver driver = new ChromeDriver();

//Goto guru99 site

driver.get("http://demo.guru99.com/V4/");

//find user name text box and fill it

driver.findElement(By.name("uid")).sendKeys("Driver 3");

}

}

****TestNG.XML****

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

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name="TestSuite" thread-count="3" parallel="methods" >

<test name="testGuru">

<classes>

<class name="TestGuru99MultipleSession">

</class>

</classes>

</test>

</suite>

## What is a headless browser?

A headless browser is a term used to define browser simulation programs that do not have a GUI. These programs execute like any other browser but do not display any UI. In headless browsers, when Selenium tests run, they execute in the background. Almost all modern browsers provide the capabilities to run them in a headless mode.

#### ****How to use HtmlUnitDriver as a headless browser with Selenium?****

To implement headless testing, Selenium makes use of HtmlUnitDriver, which is another implementation of WebDriver, similar to FirefoxDriver, ChromeDriver, etc. HTMLUnitDriver is available as an external dependency and requires you to add the library explicitly.

Once you have the jar added to the Eclipse project, you can import the class "****org.openqa.selenium.htmlunit.HtmlUnitDriver****" into your test code by adding the below line -

import org.openqa.selenium.htmlunit.HtmlUnitDriver;

After that, You can create a HtmlUnitWebDriver instance as below -

HtmlUnitDriver unitDriver = new HtmlUnitDriver();

The above syntax will construct a new instance with **[JavaScript](https://www.toolsqa.com/javascript/javascript-tutorial/)** disabled, and the default browser version.

import org.openqa.selenium.htmlunit.HtmlUnitDriver;

public class headlessBrowserDemo {

public static void main(String[] args) {

// Declaring and initialising the HtmlUnitWebDriver

HtmlUnitDriver unitDriver = new HtmlUnitDriver();

// open demo site webpage

unitDriver.get("https://demoqa.com/");

//Print the title of the page

System.out.println("Title of the page is -> " + unitDriver.getTitle());

}

}

**How to perform cross-browser testing?**

Step I : Add two dependencies A] java client (Appiam) B] Selenium java

Step II: Create following class (you will get username and accessKey after loginn in browserStack)

**package** test;

**import** java.net.MalformedURLException;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.remote.DesiredCapabilities;

**import** org.openqa.selenium.remote.RemoteWebDriver;

**import** java.net.\*;

**public** **class** BrowserStack {

**public** **static** **final** String ***userName*** ="satishpawar\_pdEiSI"; // just login to browserstack and in access key header you get both username and accessKey

**public** **static** **final** String ***accessKey***="XFt6VpE6YgnK9dsqzsTq";

**public** **static** **final** String ***URL*** = "https://"+***userName***+":"+***accessKey***+"@hub-cloud.browserstack.com/wd/hub";

**public** **static** **void** main(String[] args) **throws** MalformedURLException {

DesiredCapabilities caps = **new** DesiredCapabilities();

caps.setCapability("os", "windows");

caps.setCapability("os\_version","10");

caps.setCapability("browser", "chrome");

caps.setCapability("browser\_version", "80");

caps.setCapability("name", "Search-selenium code");

WebDriver driver = **new** RemoteWebDriver(**new** URL (***URL***),caps);

driver.get("https://www.google.com");

WebElement search = driver.findElement(By.*xpath*("//input[@title='Search']"));

search.sendKeys("Satish pawar");

System.***out***.println(driver.getTitle());

driver.quit(); // while working on browserstack we must quit session.

}

}

Step III : Run above class

Step IV: Check result on browser Stack site at left top corner below DASHBOARD

