If you are reading this document, I am assuming that you have a general knowledge of Selenium and writing tests in Selenium (in JAVA). In this document I will be sharing my experiences, different issues faced while setting up the **Test Automation in Jenkins.**

So let’s start with the Automation Script.

I am assuming that everyone here knows the basic of Selenium script, so let’s jump directly to the topic.

Most of us write the code for automation in Windows operation system, execute them in the command prompt and assume that these codes will also run in the Server (Linux OS). This is not the case.

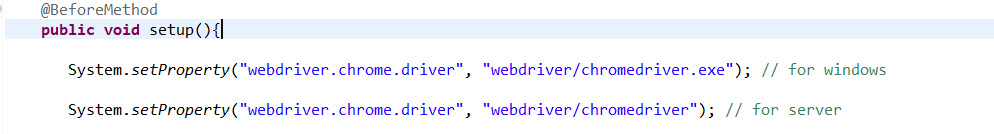
The things which needs to be changed for Linux are:

**WebDrivers**

First of all, the web drivers that run on Windows won’t run on Server because the server doesn’t support .exe format.

We need to download separate web drivers for Linux. Which can be easily found online.

Then the path for the web drivers in the Setup Class also need to be changed.



**Code**

We just need to add some extra lines of code in the setup class for the script to run in the server.We need to use ChromeOptions calss to manage options sepcific to chrome driver. Simply what ChromeOptions does is, it adds additional command line arguments to be used when starting Chrome.



As shown in the code above, we have to create an instance of ChromeOptions to set ChromeDriver specific capabilities and then pass the ChromeOptions object to the WebDriver constructor.

chromeOptions.addArguments("headless");

chromeOptions.addArguments("disable-gpu");

chromeOptions.addArguments("window-size=1200,1100");

chromeOptions.addArguments("--no-sandbox");

chromeOptions.addArguments("disable-infobars");

There are the improtant arguments that need to be used to run the code in Server.

If you want to learn more about the ChromeOptions Class. You can go to this [LINK](https://seleniumhq.github.io/selenium/docs/api/java/org/openqa/selenium/chrome/ChromeOptions.html).

After we make these changes in the code, we can push it to Git so that it can be pulled to the Server.

But before doing that there are certain things that need to be done in the Server first. Which are:

1. **Install Chrome**

Chrome shoulde be installed in the server for the automation script to run successfully. The steps to isntall chrome are as follows:

1. sudo curl -sS -o - https://dl-ssl.google.com/linux/linux\_signing\_key.pub | apt-key add
2. sudo echo "deb http://dl.google.com/linux/chrome/deb/ stable main" >> /etc/apt/sources.list.d/google-chrome.list
3. sudo apt-get -y update
4. sudo apt-get -y install google-chrome-stable
5. **Install ChromeDriver:**

Similarly, ChromeDriver must also be installed in the server. The steps to install ChromeDriver are as follows:

1. wget –N http://chromedriver.storage.googleapis.com/$CHROME\_DRIVER\_VERSION/chromedriver\_linux64.zip -P ~/
2. unzip ~/chromedriver\_linux64.zip -d ~/
3. rm ~/chromedriver\_linux64.zip
4. sudo mv -f ~/chromedriver /usr/local/bin/chromedriver
5. sudo chown root:root /usr/local/bin/chromedriver
6. sudo chmod 0755 /usr/local/bin/chromedriver

After we install Chrome and ChromeDriver in the server we are good to go for running the automtion code in the server.

First we need to Clone the Git repo of the Automation in the Server.

Then the process after that is Explained below:

1. **Compiling the code from terminal:**

We use different IDE like Eclipse, intellij, NetBeans etc to write our code, althought the IDE compiles the code automatically, the structure of the bin folder is different for different IDE which will casue problem when giving the path of the java files inside bin, so we must compile the code in the server during execution.

The process to complie the code is described follows:

We have a standard java project that consists of three top level folders:

**/bin** - empty folder that will contain compiled .class files

**/lib** - contains third party .jar files

**/src** - contains .java source files

Assuming we are at the application root folder trying to compile **SetupAndTeardown.java** file from package **test.BackendCMS.Test** in the src folder that uses lib1.jar and lib2.jar libraries from lib folder to a destination bin folder, compilation command should have the following format:

**Code:** javac -d bin -sourcepath src -cp lib/lib1.jar:lib/lib2.jar ./src/test/BackendCMS/Test/SetupAndTeardown.java

As a result bin/ test/BackendCMS/Test/SetupAndTeardown.class file should be created.

This is just an example to make up understand the code.

The tipical code to compli the code will look like:

Code: javac -d bin -sourcepath src -cp lib/selenium-chrome-driver-3.11.0.jar:lib/exceldataprovider.jar:lib/selenium-server-standalone-3.11.0.jar:lib/testng-6.14.2.jar ./src/test/BackendCMS/Test/\*

This code will compile all the .java files in src/test/BackendCMS/Test folder and store them in the bin folder.

1. **Executing Automation in Server:**

After we have finished all the setups mentioned above now it time to run the automation.

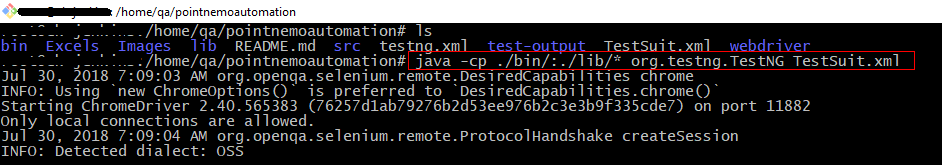
Below are the steps you need to follow:

1. You must be in the director where the bin, lib, src folder etc are located.
2. Then you enter the code

Code: java -cp ./bin/:./lib/\* org.testng.TestNG TestSuit.xml

[Note: Make sure that the .xml file is present.]

After this the automation script will be executed in the terminal.



Here the project directory is pointnemoautomation and the name of the .xml file is TestSuit.xml

Let’s start with the introduction to **Jenkins:**

Jenkins is till now the most widely used tool for [Continuous Integration.](https://en.wikipedia.org/wiki/Continuous_integration)

Jenkins is a self-contained, open source automation server which can be used to automate all sorts of tasks related to building, testing and delivering or deploying software.