



XSLT Reporting

Key Points:

Agenda

- ▶ What is XSLT Report (XML Style-sheet language for transformation)
- ▶ How to use ANT build tool
- ▶ Download ANT
- ▶ Verify ANT Installation
- ▶ Execute Build
- ▶ Verify report folder

XSLT Report

XSLT stands for XML Style-sheet language for transformation, it provide very rich formatting report using TestNG framework

To generate XSLT report in Selenium is ready with below precondition.

Precondition-

1- Ant should be installed-

2- We should have some test case should be executed by TestNG (i.e. -output directory should be available in home directory)

Install ANT-



Install ANT

What is Ant?

1-Apache Ant is a Java based build tool from Apache.

2-Apache Ant's build files are written in XML .

3-Open Source tool

This post will cover you how to Install Apache ANT to automate the build and deployment process in simple and easy steps.

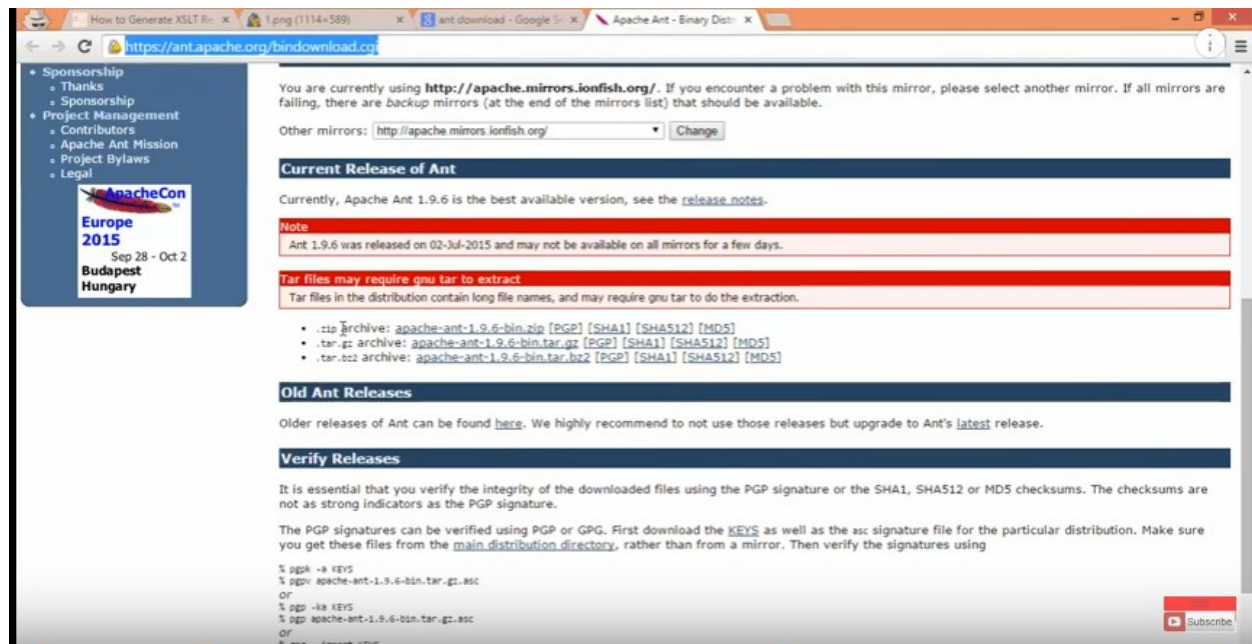
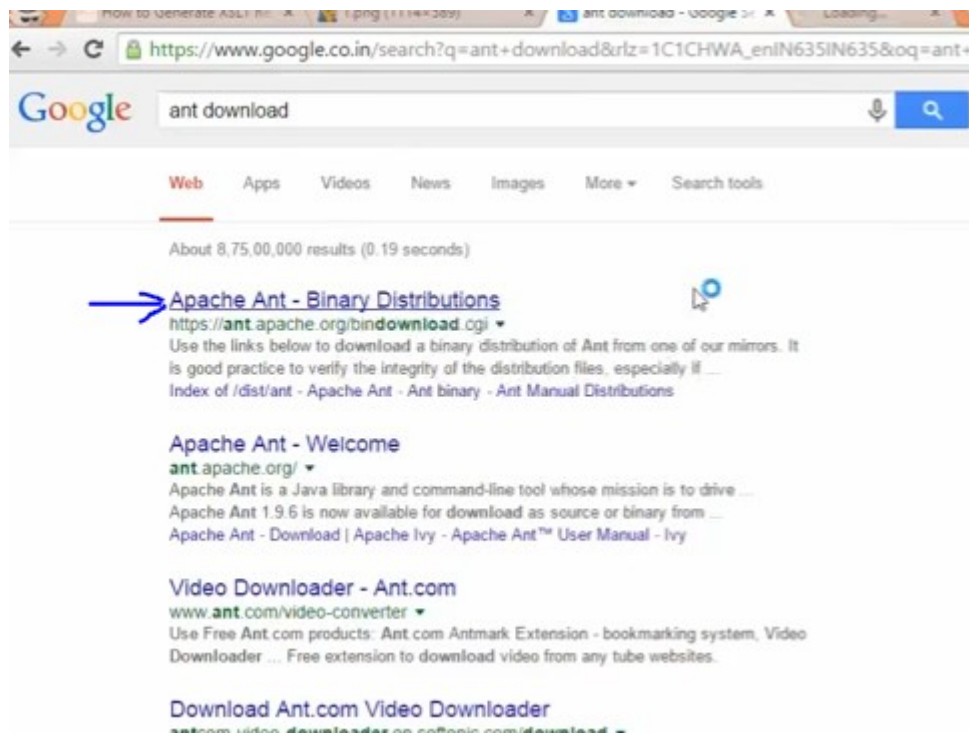
Let us Generate XSLT Report in Selenium

Step 1- Navigate to below mention url

<http://ant.apache.org/bindownload.cgi>

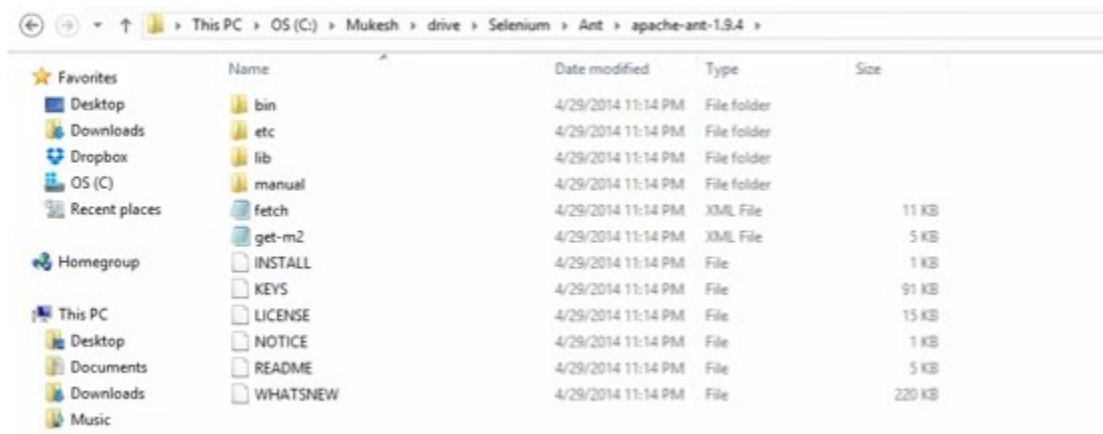
Step 2- Navigate to Current release of ant and download the zip file

Download Ant

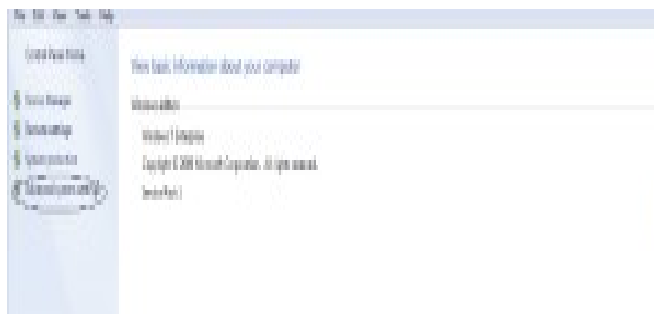


Generate XSLT report in Selenium

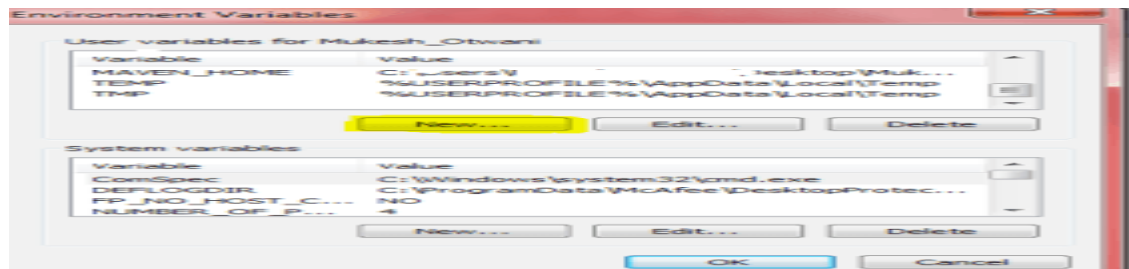
Step 3- Extract zip file - Example



Step 4- Once we extract the zip file then we need to set environment variable
Right click on My computer and select the properties and click on Advanced system setting

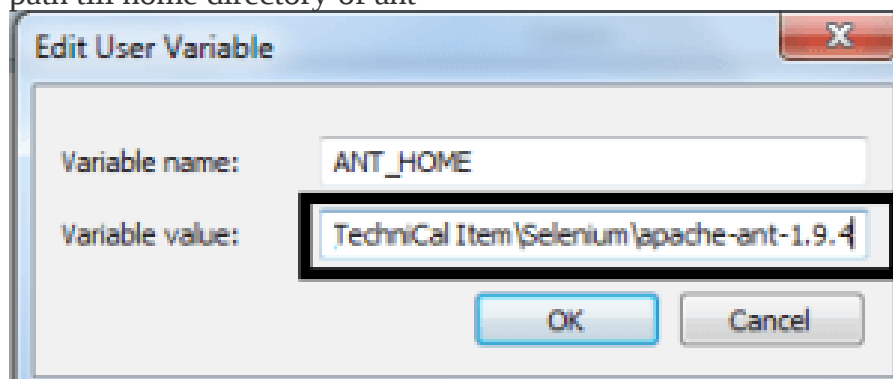


System setting



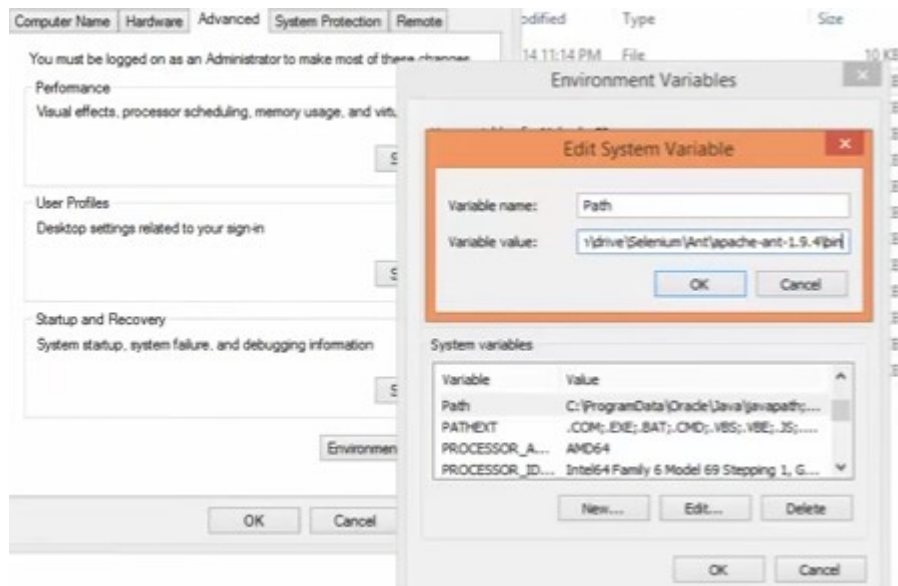
Add new variable

Add the user variable – Here give the name ANT_HOME and in value section specify the path till home directory of ant



Latest is 1.9.6 – (Ant)

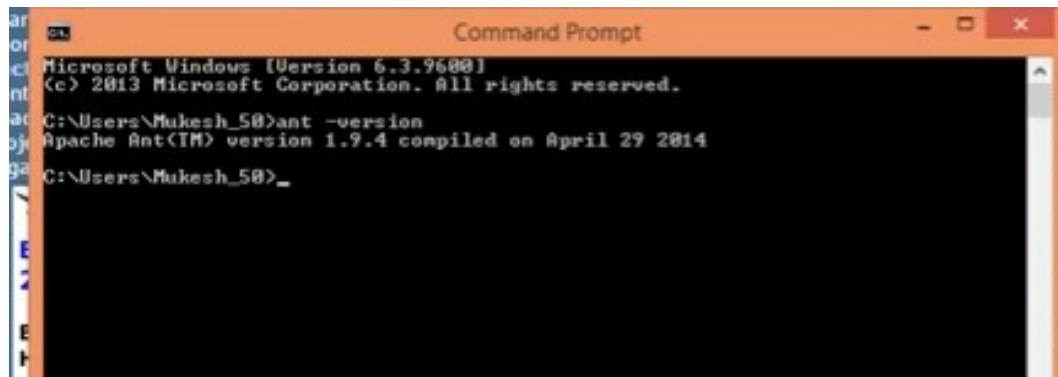
Add system variable- In this we need to edit existing system path click on edit, go till last and give the location till bin and save then semicolon.



Generate XSLT Report

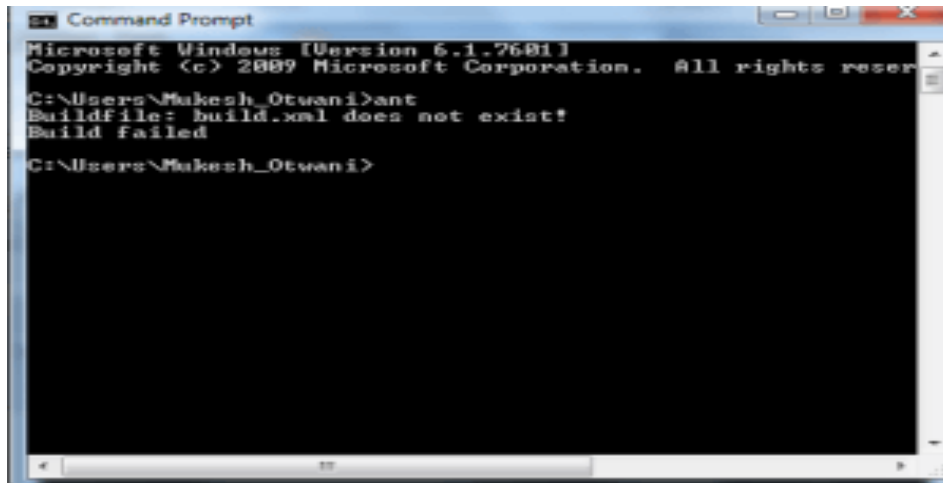
Note- Please do not edit other path- it may crash your system.

Step 5- Now verify that Ant is installed properly- Open CMD and type run and hit enter.



Ant is installed

Note- if it is not installed properly then in output console we will get build.xml not found- build failed the restart your system.

A screenshot of a Windows Command Prompt window. The title bar reads "Command Prompt". The window content shows the following text: "Microsoft Windows [Version 6.1.7601] Copyright (c) 2009 Microsoft Corporation. All rights reserved." followed by the command "C:\Users\Makesh_Otwani>ant". The output of the command is "Buildfile: build.xml does not exist!" and "Build failed". The prompt "C:\Users\Makesh_Otwani>" is visible at the bottom of the window.

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Makesh_Otwani>ant
Buildfile: build.xml does not exist!
Build failed

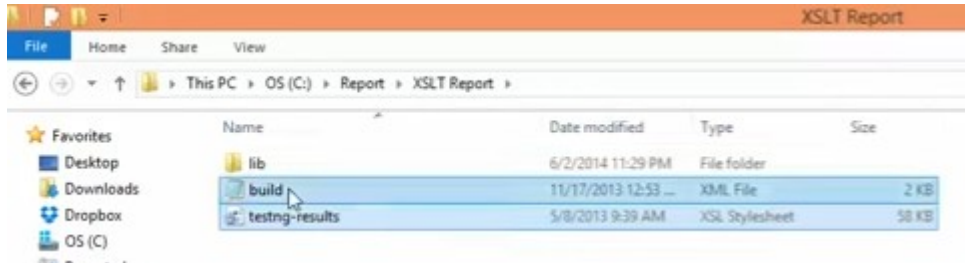
C:\Users\Makesh_Otwani>
```

Build failed

XSLT

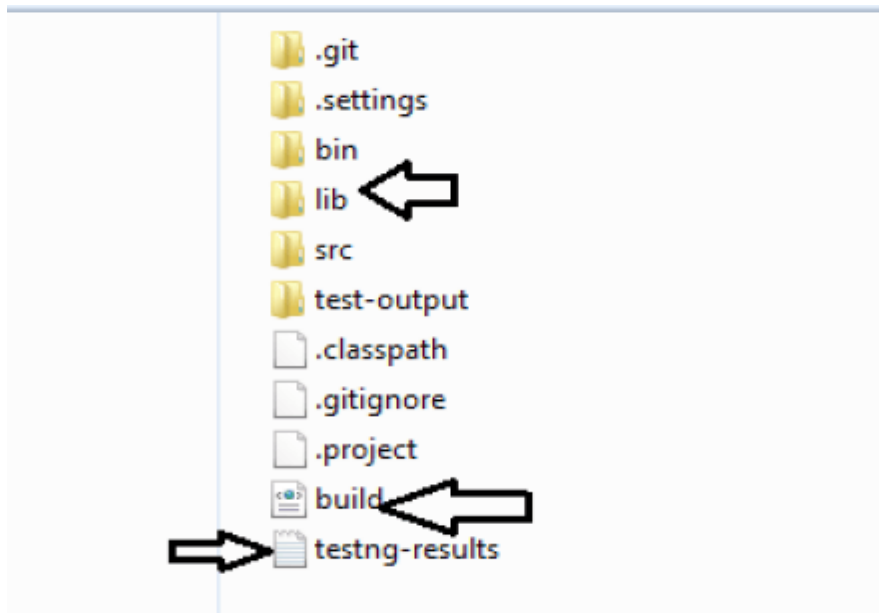
Step 1- [Download XSLT from my Google driver account.](#)

Step 2- Unzip XSLT Folder and copy all the files and paste into project home directory.
Refer below screen-shot



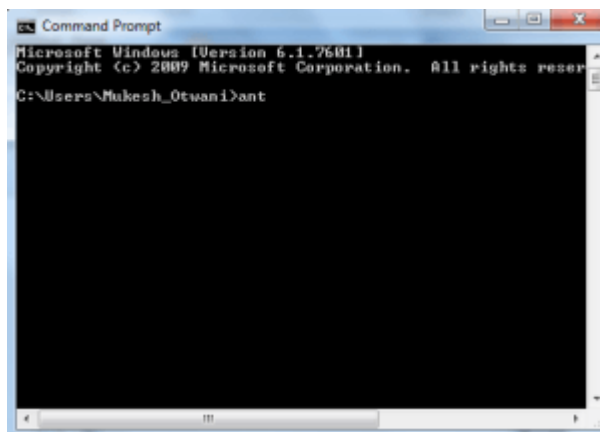
Name	Date modified	Type	Size
.git	6/1/2014 2:07 AM	File folder	
.settings	6/1/2014 1:49 AM	File folder	
bin	6/1/2014 2:36 PM	File folder	
src	6/1/2014 1:57 AM	File folder	
.classpath	6/1/2014 1:49 AM	CLASSPATH File	1 KB
.gitignore	6/1/2014 1:57 AM	GITIGNORE File	1 KB
.project	6/1/2014 1:49 AM	PROJECT File	1 KB

Download XSLT Folder

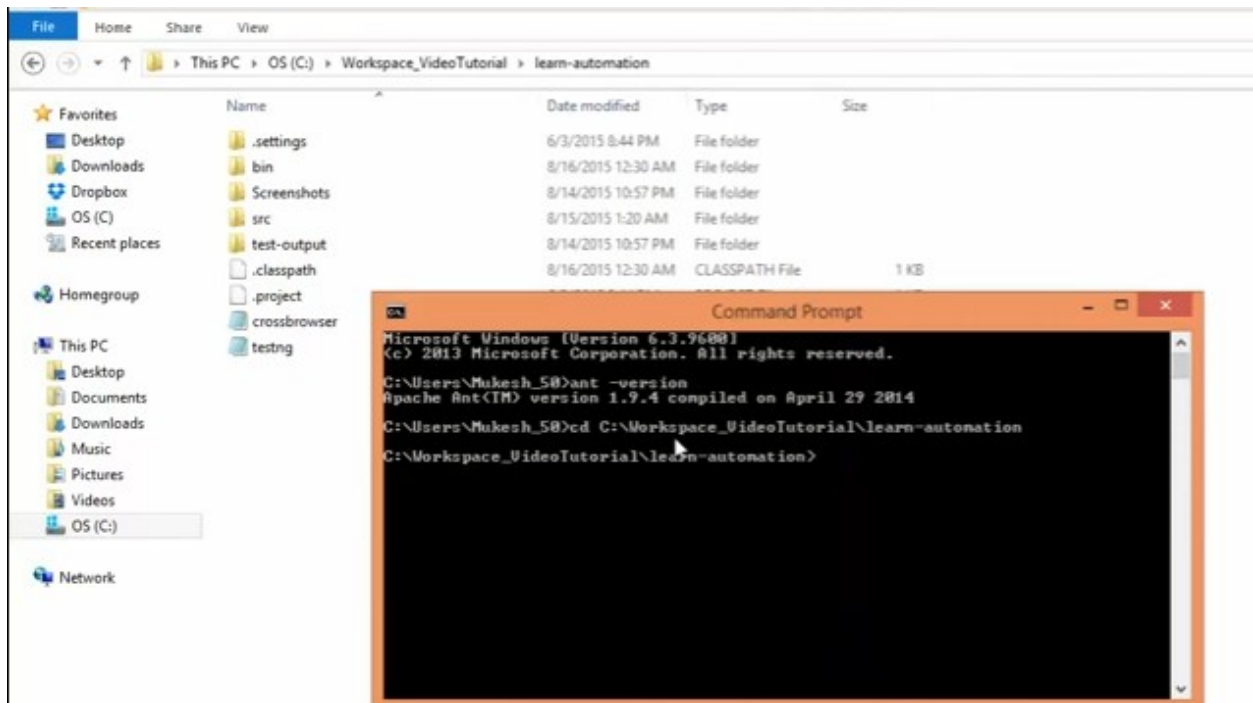


Unzip and add the XSLT Folder

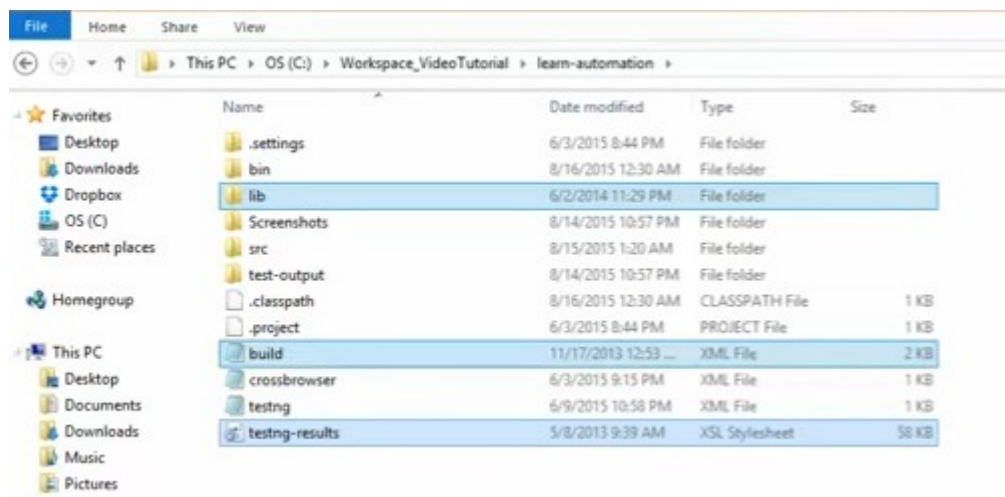
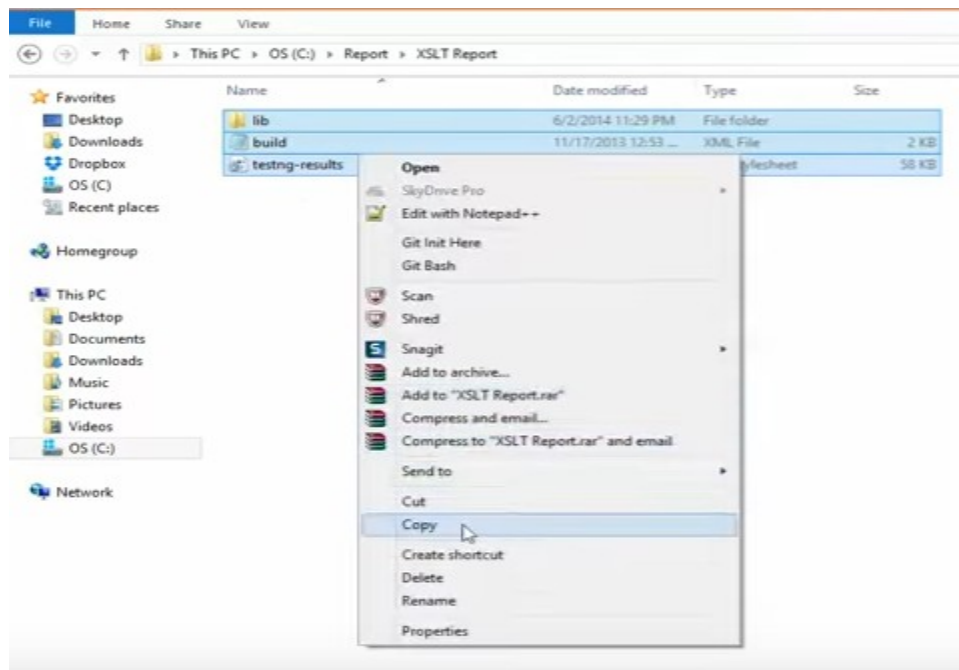
Step 3- Now execute build.xml file using Ant – for running build.xml file
Open command prompt and go till project home directory and type run and hit enter.



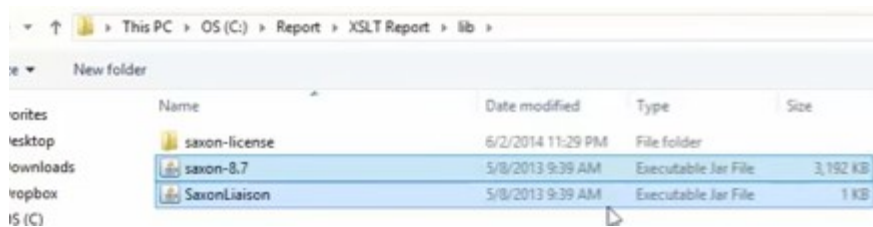
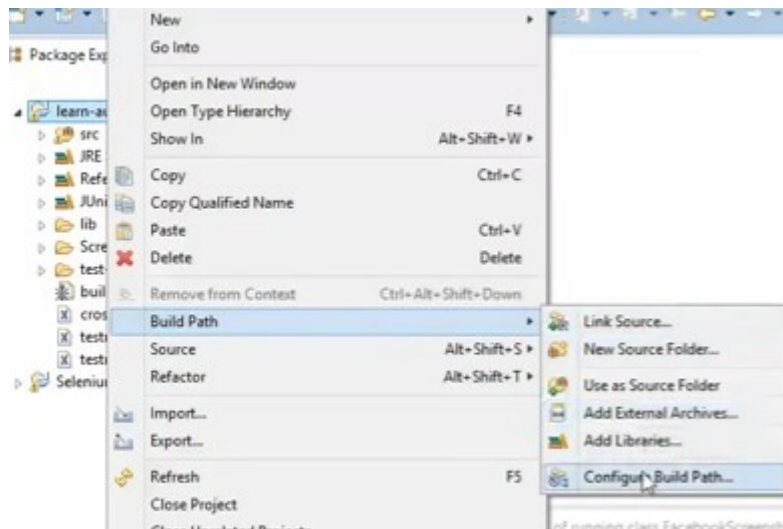
Home directory- specify path of project in a workspace



Step 4 - Copy xslt-report contents into workspace



Step 5- Go to eclipse configure build path -> Add external Jar, Add xslt jar file



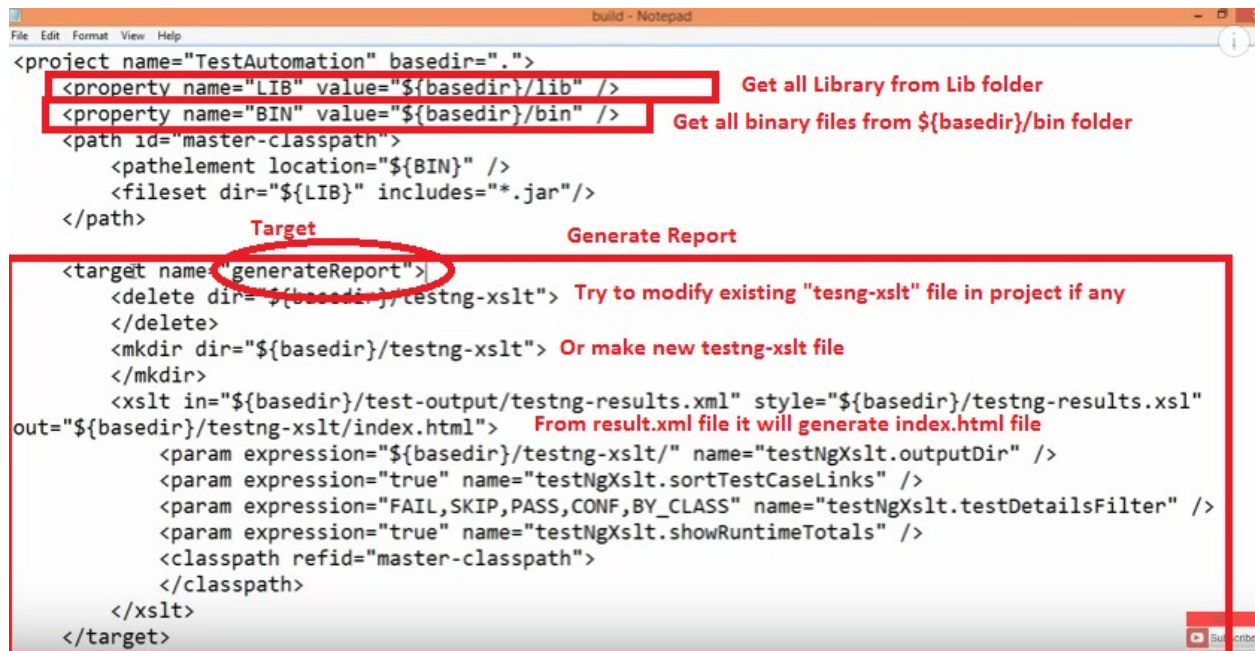
Step 6- Open command Prompt and type **ant**

```
C:\Workspace_VideoTutorial\learn-automation>ant
Buildfile: C:\Workspace_VideoTutorial\learn-automation\build.xml

BUILD SUCCESSFUL
Total time: 1 second

C:\Workspace_VideoTutorial\learn-automation>
```

Step 7- Go to Build.xml file from workspace → project.



```
<project name="TestAutomation" basedir=".">
  <property name="LIB" value="${basedir}/lib" />
  <property name="BIN" value="${basedir}/bin" />
  <path id="master-classpath">
    <pathelement location="${BIN}" />
    <fileset dir="${LIB}" includes="*.jar"/>
  </path>
  <target name="generateReport">
    <delete dir="${basedir}/testng-xslt">
    </delete>
    <mkdir dir="${basedir}/testng-xslt">
    </mkdir>
    <xslt in="${basedir}/test-output/testng-results.xml" style="${basedir}/testng-results.xsl"
out="${basedir}/testng-xslt/index.html">
    <param expression="${basedir}/testng-xslt/" name="testNgXslt.outputDir" />
    <param expression="true" name="testNgXslt.sortTestCaseLinks" />
    <param expression="FAIL,SKIP,PASS,CONF,BY_CLASS" name="testNgXslt.testDetailsFilter" />
    <param expression="true" name="testNgXslt.showRuntimeTotals" />
    <classpath refid="master-classpath">
    </classpath>
  </xslt>
</target>
```

Get all Library from Lib folder

Get all binary files from \${basedir}/bin folder

Target

Generate Report

Try to modify existing "testng-xslt" file in project if any

Or make new testng-xslt file

From result.xml file it will generate index.html file

Build now

Step 8 - Once build is successful then write **ant generateReport** and hit enter.

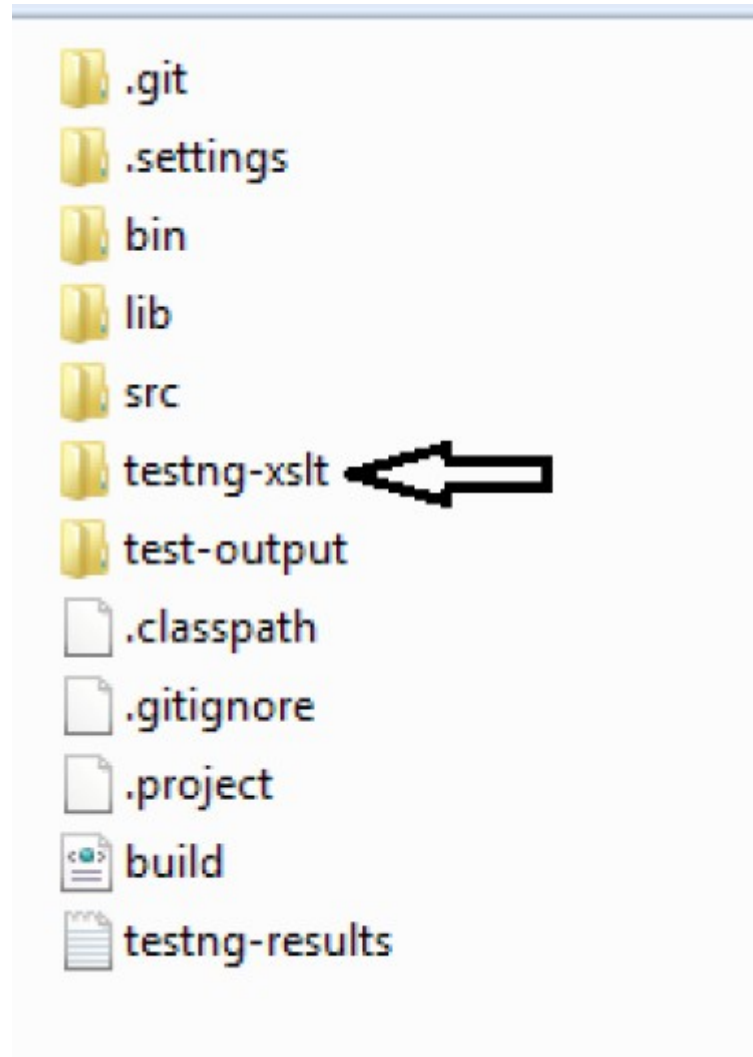


Generate XSLT Report

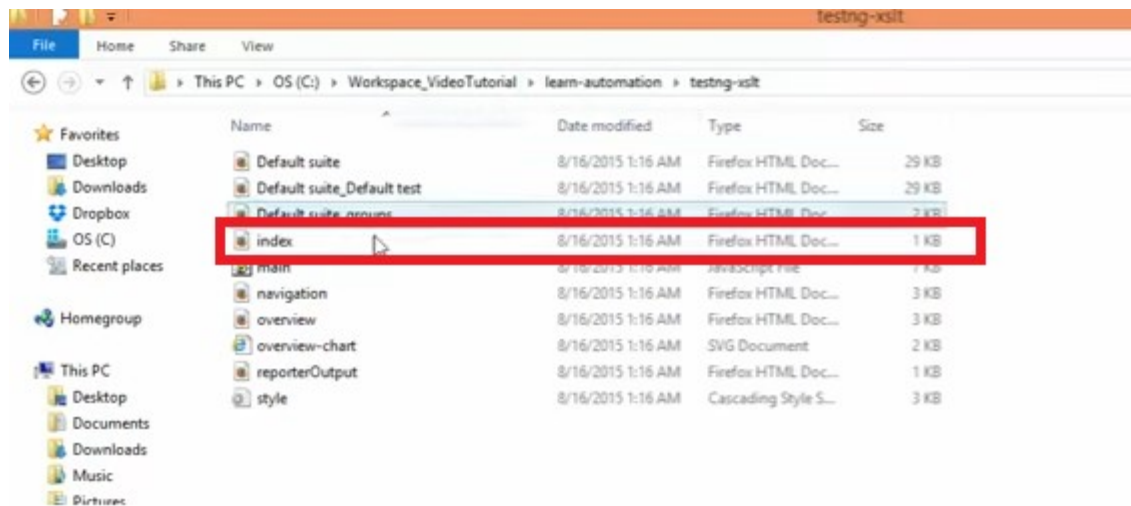
```
BUILD SUCCESSFUL
Total time: 1 minute 35 seconds
```

Generate XSLT Report

Step 9 - After build is successful navigate to project directory and you will get testng-xslt folder

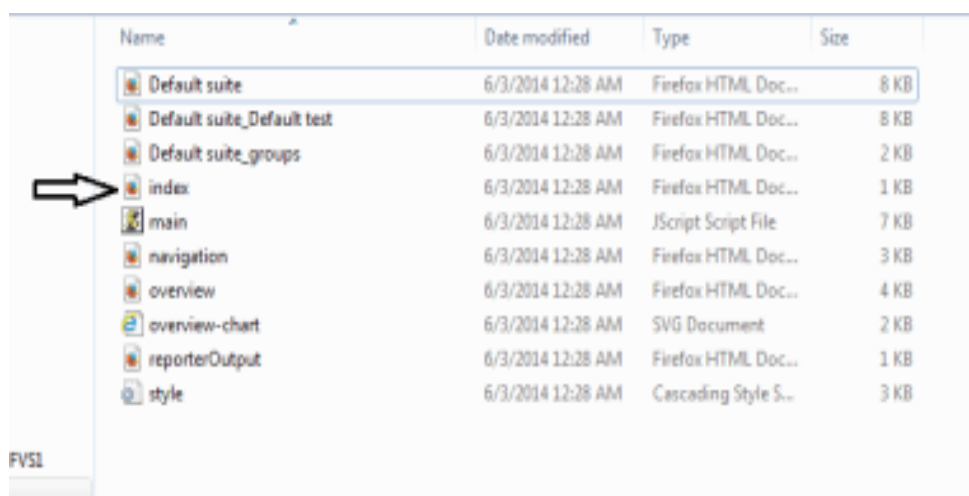


Step 10 – Open index.html from testng-xslt



Generate XSLT Report

Inside testng-xslt you will get index.html (this is the main report) open in Firefox or in Chrome browser which support JavaScript



Generate XSLT Report

Test suites overview

Failed (20%)
Passed (57%)
Skipped (14%)



Wipro	2	3	0	5	60%	406 ms
InstallationDemo	2	3	0	5	60%	406 ms
IBM	4	9	2	15	60%	64 ms
InstallationDemo	4	9	2	15	60%	64 ms
TCS	2	0	2	4	0%	129 ms
InstallationDemo	2	0	2	4	0%	129 ms
Inf	12	27	6	45	60%	27 ms
Release 10:01	4	9	2	15	60%	22 ms
Release 10:06	4	9	2	15	60%	20 ms
Release 10:00	4	9	2	15	60%	20 ms

Generate XSLT Report

☒ Failed ☒ Passed ☒ Skipped ☒ Config

TestRegistrationSmoke			
Name	Started	Duration	Exception
Test01	10/14/22	0:04	java.lang.AssertionError: expected[0:0:700] but found[0:0:7]
Test02	10/14/22	1:04	java.lang.AssertionError: expected[0:0:700] but found[0:0:7]
Test03	10/14/22	0:04	
Test04	10/14/22	0:04	
Test05	10/14/22	1:04	
TestRegistrationSmoke			
Name	Started	Duration	Exception
Test01	10/14/22	0:04	java.lang.AssertionError: expected[0:0:700] but found[0:0:7]
Test02	10/14/22	1:04	java.lang.AssertionError: expected[0:0:700] but found[0:0:7]
Test03	10/14/22	0:04	
Test04	10/14/22	1:04	
Test05	10/14/22	0:04	
TestRegistrationSmoke			
Name	Started	Duration	Exception
Test01	10/14/22	0:04	java.lang.AssertionError: expected[0:0:700] but found[0:0:7]
Test02	10/14/22	0:04	java.lang.AssertionError: expected[0:0:700] but found[0:0:7]
Test03	10/14/22	0:04	
Test04	10/14/22	1:04	
Test05	10/14/22	0:04	
TestRegistrationSmokeTest			
Name	Started	Duration	Exception
SmokeTest	10/14/22	0:04	