Developer WorkBench for JEE

Lab Book

Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Revision No.** | **Author** | **Summary of Changes** |
| 01-Nov -11 | 1.0 | Rathnajothi Perumalsamy |  |
| 5-6-2016 | 1.1 | Zainab | ANT was replaced with MAVEN |
|  |  |  |  |

Table of Contents

[Document Revision History 2](#_Toc451044065)

[Table of Contents 3](#_Toc451044066)

[Getting Started 4](#_Toc451044067)

[Overview 4](#_Toc451044068)

[Setup Checklist for Maven and PMD 4](#_Toc451044069)

[Instructions 4](#_Toc451044070)

[Lab 1. Getting Started With Maven 5](#_Toc451044071)

[1.1: Setting environmental variables 5](#_Toc451044072)

[1.2 Configuring Maven Settings: 8](#_Toc451044073)

[1.3 Creating first standalone Maven application using archetype: 9](#_Toc451044074)

[Figure 8: Creating first standalone Maven application using archetype 10](#_Toc451044075)

[<<TODO>> 12](#_Toc451044076)

[Lab 2. Integrating Maven with Eclipse 13](#_Toc451044077)

[2.1: Configure environment to run Maven project 13](#_Toc451044078)

[2.2: Creating a sample Maven project 13](#_Toc451044079)

[<<TODO>> 19](#_Toc451044080)

[Lab 3. PMD Tool 20](#_Toc451044081)

[2.1: Installing PMD 20](#_Toc451044082)

[2.2 Using PMD with Eclipse 20](#_Toc451044083)

[<< TO DO>> 23](#_Toc451044084)

[Appendices 24](#_Toc451044085)

[Appendix A: Table of Figures 24](#_Toc451044086)

Getting Started

## Overview

This lab book is a guided tour for learning Maven tool and PMD tool. It comprises step by step guidance and ‘To Do’ assignments. Follow the steps provided in the solved examples and work out the ‘To Do’ assignments given which will expose you to working with Java applications.

## Setup Checklist for Maven and PMD

Here is what is expected on your machine in order for the lab to work.

Minimum System Requirements

* Intel Pentium 90 or higher (P166 recommended)
* Microsoft Windows 95, 98, or NT 4.0, 2k, XP.
* Memory: 32MB of RAM (64MB or more recommended)
* Internet Explorer 6.0 or higher
* MS-Access/Connectivity to Oracle database
* Apache Tomcat Version 5.0.
* Apache Maven 3.0

Please ensure that the following is done:

* Eclipse Luna is installed.
* JDK 1.8 is installed. (This path is henceforth referred as <java\_install\_dir>)
* Wildfly 8.1

## Instructions

* For all coding standards refer Appendix A. All lab assignments should refer coding standards.
* Create a directory by your name in drive <drive>. In this directory, create a subdirectory java\_assgn. For each lab exercise create a directory as lab <lab number>.

1. Getting Started With Maven

|  |  |
| --- | --- |
| **Goals** | * Learn and Understand the process of * Setting environment variables * Configuring Maven settings * Creating a simple Maven Project using commands |
| **Time** | 60 minutes |

## 1.1: Setting environmental variables

**Step1:** set M2\_Home to Maven Installation Directory using the following command:

* **set** M2**\_HOME= C:\apache-maven-version.**

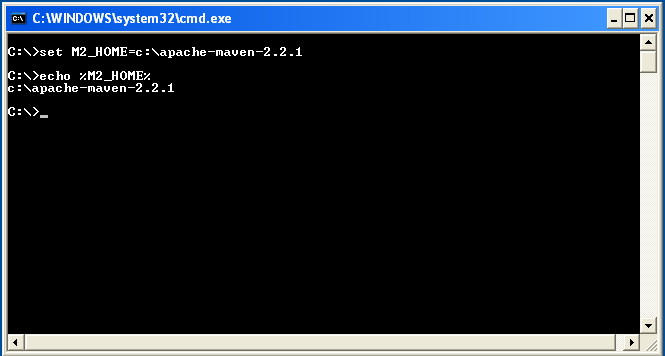


Figure 1: Environmental Variable

**Step 2:** Set **JAVA\_HOME** to Jdk1.8.0\_31 using the following command:

**i.e. JAVA\_HOME= C:**\Program Files\Java\jdk1.8.0\_31

**Step 3:** Set PATH environment variable:

* **Set PATH=%PATH%;%JAVA\_HOME%\bin;%M2\_HOME%\bin;**

**** Alternatively follow the following steps for setting the environment variables

**Alternate approach:**

**Step 1:** Right click **My Computers**, and select **Properties** 🡪 **Environment Variables**.

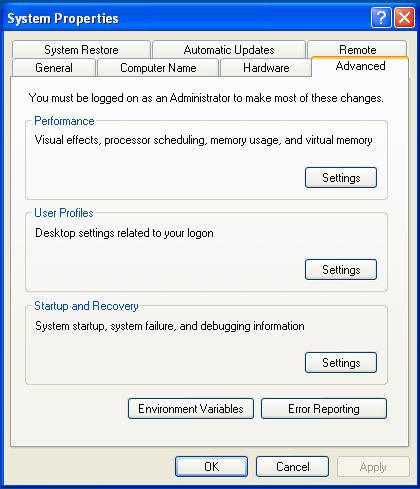


Figure 2: System Properties

**Step 2:** Click **Environment Variables**. The Environment Variables window will be displayed.

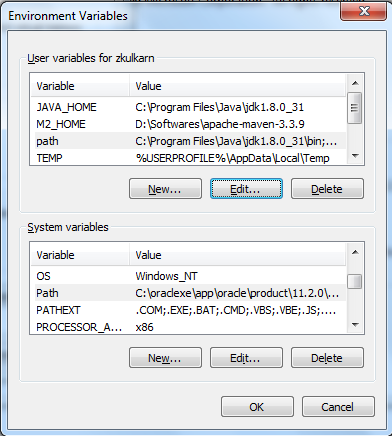
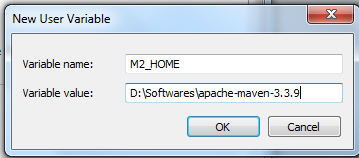


Figure 3: Environment Variables

**Step 3:** Create a new user variable M2\_HOME by clicking on edit and set the path of Apache Maven installation path as shown in the figure.



**Figure 4: Edit User Variable**

**Step 4:** Click **JAVA\_HOME** System Variable if it already exists, or create a new one and set the path of JDK1.8.0\_31 as shown in the figure.

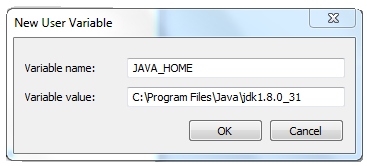
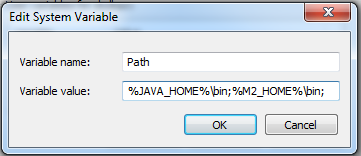


Figure 5: Edit User Variable

**Step 5:** Click **PATH** System Variable and set it as **%*JAVA\_HOME%\bin*;%M2\_HOME%\bin;**

****

**Figure 6: Edit User Variable**

## 1.2 Configuring Maven Settings:

**Step 1:** Edit settings.xml to configure the proxy settings for downloading artifacts from remote repository.

**Note:** Ignore this step if you are using local repository.

<settings>

….

<proxies>

<proxy>

<active>true</active>

<protocol>http</protocol>

<host>proxy.mycompany.com</host>

<port>8080</port>

<username>your-username</username>

<password>your-password</password>

</proxy>

</proxies>

…

</settings>

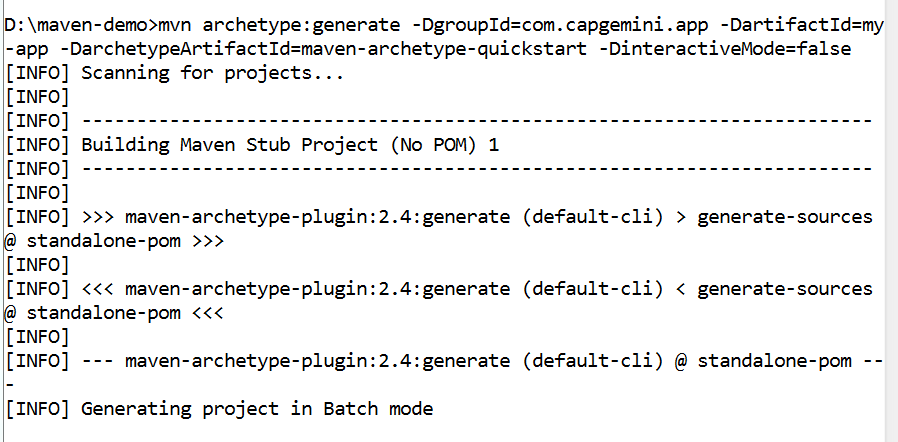
Figure 7: settings.xml

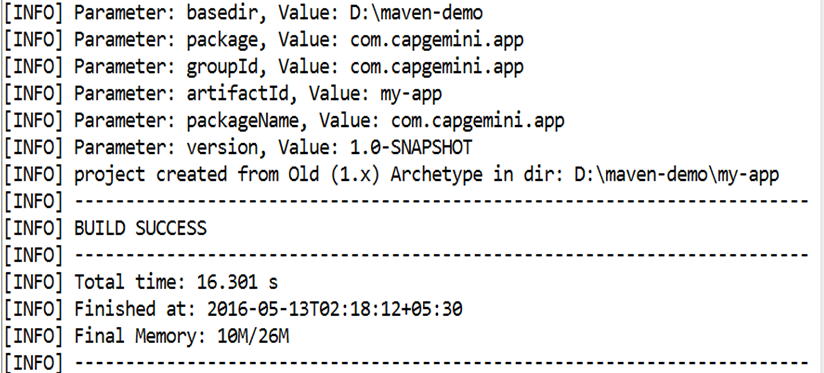
## 1.3 Creating first standalone Maven application using archetype:

**Step 1:** Execute the following command to create Maven Project

mvn archetype: generate -DgroupId=com.capgemini.app -DartifactId=myApp -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

The execution of the above command should result into the display of archetypes as shown below:





## Figure 8: Creating first standalone Maven application using archetype

**Step 2:** After successful execution of the command, Maven will create the project directory my-app having pom.xml with the contents shown below.

Keep the application specific files in ${basedir}/src/main/java and test sources reside in ${basedir}/src/test/java, where ${basedir} represents the directory containing pom.xml.

<project>

<modelVersion>4.0.0</modelVersion>

<groupId>com.capgemini.app</groupId>

<artifactId>my-app</artifactId>

<packaging>jar</packaging>

<version>1.0</version>

<name>my-app</name>

<url>http://maven.apache.org</url>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.11</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

Figure 9: Sample pom.xml

**Step 3:** Execute the "mvn compile" command to compile your application sources as shown below:

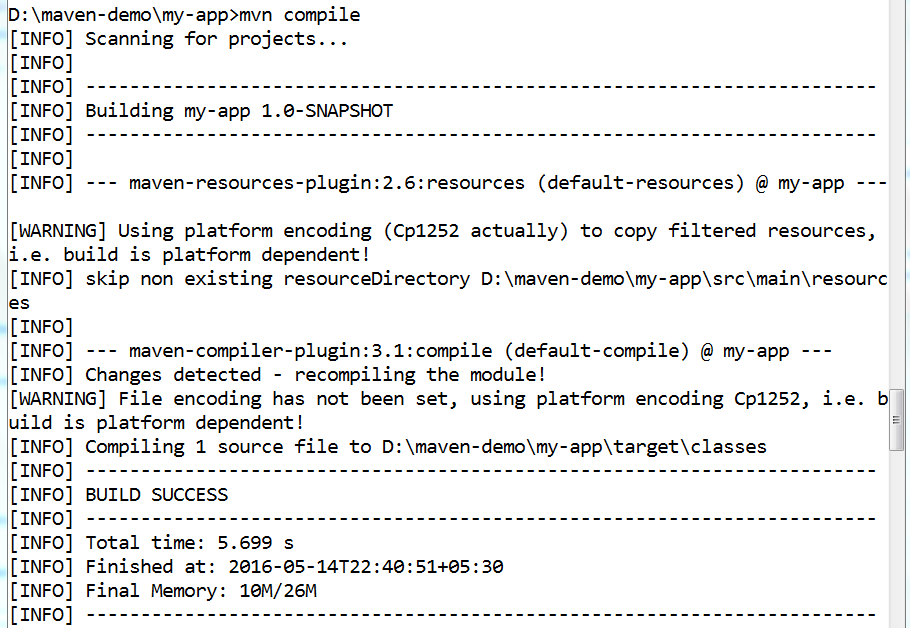


Figure 10: Compiling Sample Project

**Step 4:** Unit testing of the project can be performed by executing the following command:

**mvn test**

After successful execution of the command, the test result will be displayed which comprises the details such as test run,failures, errors...

**Step 5:** Artifact can be packaged and installed into local repository using commands such as

mvn package and mvn install.

* mvn package packages the artifact based on the packaging type specified in the pom.xml.
* mvn install installs the packaged artifact into the local repository.

**Step 6:** Basic Standard site for project can also be created using command mvn site.

## 

## <<TODO>>

**Assignment-1:** Create a Banking System project in maven which maintains two kinds of accounts for customers, one called savings account and the other as current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book but no interest. Current account holders should have a minimum balance else they should pay service charges. Build, test and deploy the project into local repository.

1. Integrating Maven with Eclipse

|  |  |
| --- | --- |
| **Goals** | * Learn and Understand the process of * Installing m2eclipse plugin * Configure environment to run Maven Project * Creating a simple Maven Project in eclipse |
| **Time** | 60 minutes |

## 2.1: Configure environment to run Maven project

**Step1:** To execute Maven project, eclipse needs to be run using JDK instead of JRE.

**Step2:** Start the eclipse now, tocreate a Maven Project and run successfully.

## 2.2: Creating a sample Maven project

Create a simple java project named ‘myproject’.

Solution:

**Step 1:** Open **eclipse3.3**.

**Step 2:** Select **File** 🡪 **New** 🡪 **Project** 🡪 **Maven project**.

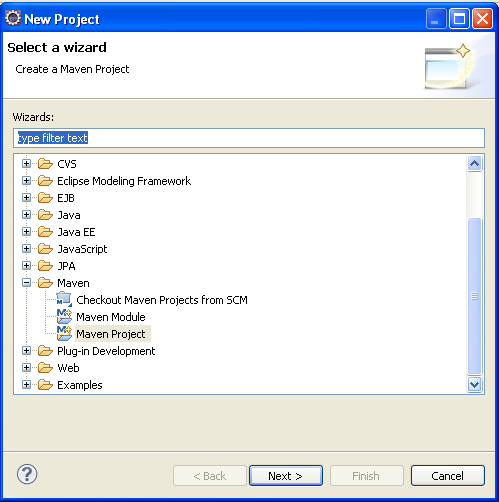


Figure 11: Select Wizard in Eclipse

**Step 3:** Choose Maven Project and use the default Workspace location or specify the location if necessary.

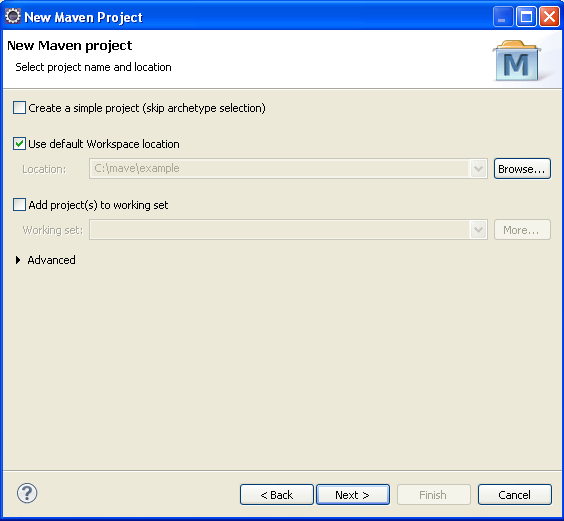


Figure 12: New Maven Project

**Step 4:** Select the maven-archetype-quickstart archetype from the list.

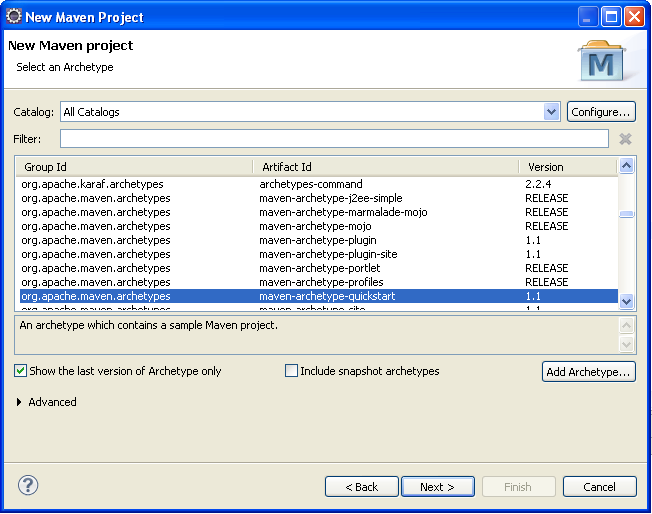


Figure 13: Selecting an archetype

**Step 5:** Enter the project coordinate details such as Group Id, Artifact Id and click 'Finish'

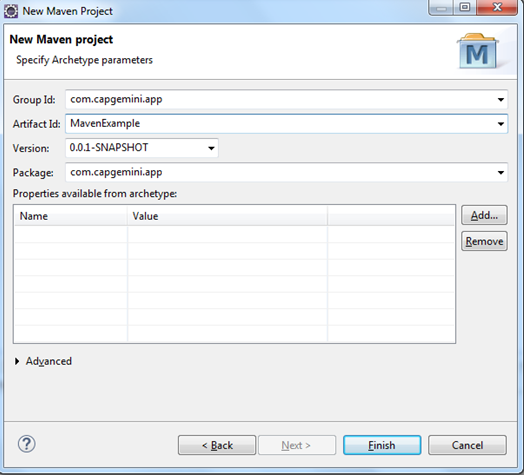


Figure 14: Specifying Archetype Parameters

**Step 6:** To build the project, right click on project named “examples” and choose Maven Build under Run As option.

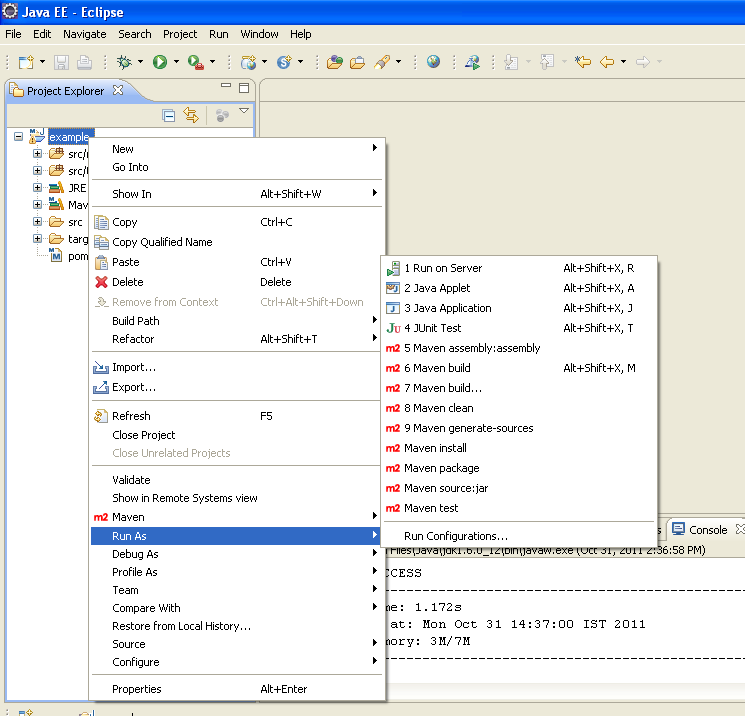


Figure 15: Building an Application

## <<TODO>>

**Assignment 1:** Create aweb application which displays product details such as Product Name, Product description, and its price. Users can place orders specifying the quantity of each product. Once the order is placed by customer, the invoice for the current products transaction showing the product name, quantity ordered, price and total amount should be displayed. Build and execute the common life cycle phases for the web application in eclipse.

**Assignment 2:** Use the Assignment 1 in Lab 1, import the Banking system project into eclipse and build the project in eclipse environment.

1. PMD Tool

|  |  |
| --- | --- |
| **Goals** | At the end of this lab session, you will be able to:   * + Install PMD.   + Use PMD with Eclipse. |
| **Time** | 90 minutes |

## 2.1: Installing PMD

Solution:

**Step 1:** Download the latest binary distribution - i.e., pmd-bin-x.xx.zip from Http://pmd.sourceforge.net/index.html or obtain it from the faculty.

**Step 2**: Extract the contents of the downloaded zip file.

## 2.2 Using PMD with Eclipse

Solution:

#### Step 1: import the Project named PMDExample.

**Step 2:** select the project and click properties. Select PMD and check Enable PMD

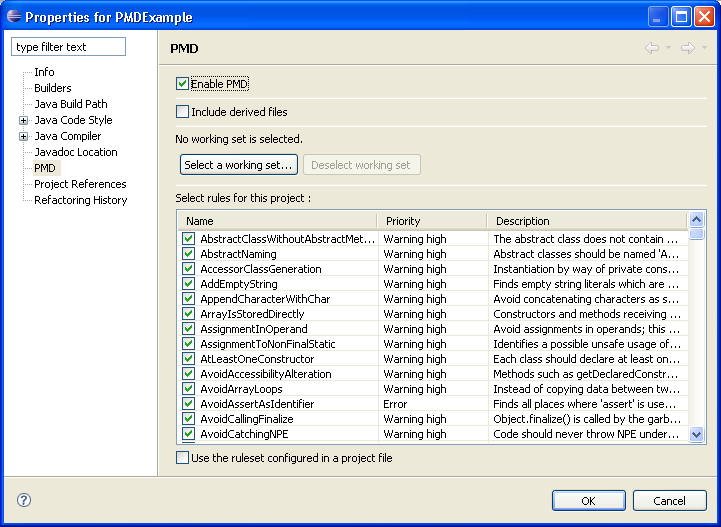


Figure 16: Enable PMD

Step 3: To run PMD with Eclipse right click on the project, select PMD, check code with PMD.

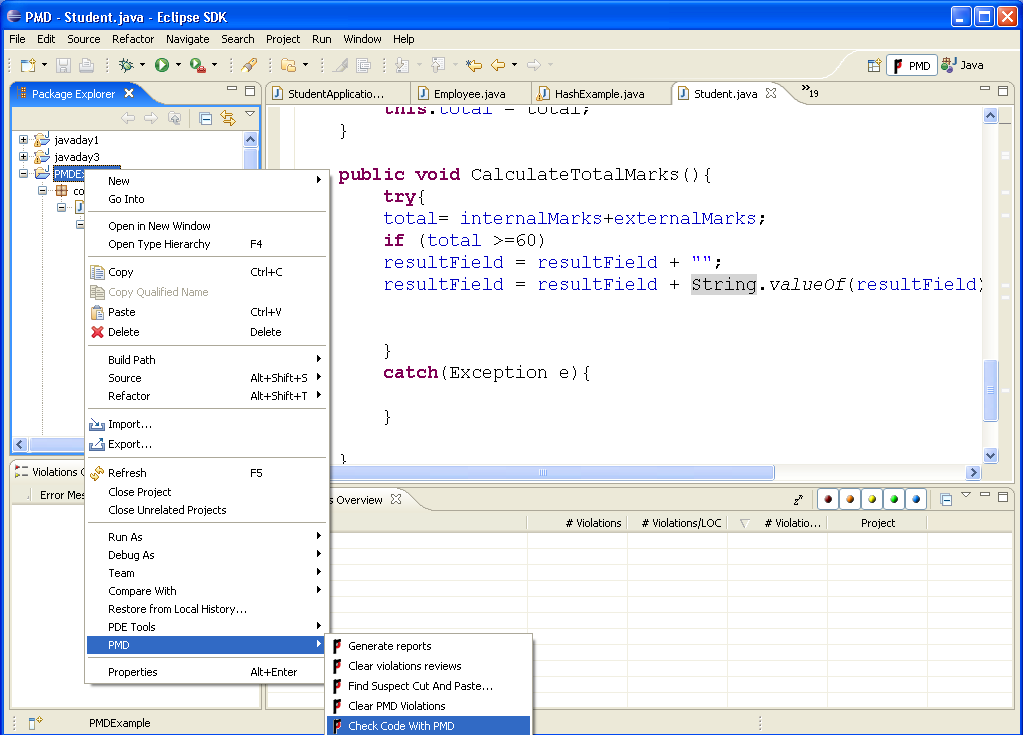


Figure 17: check code with PMD

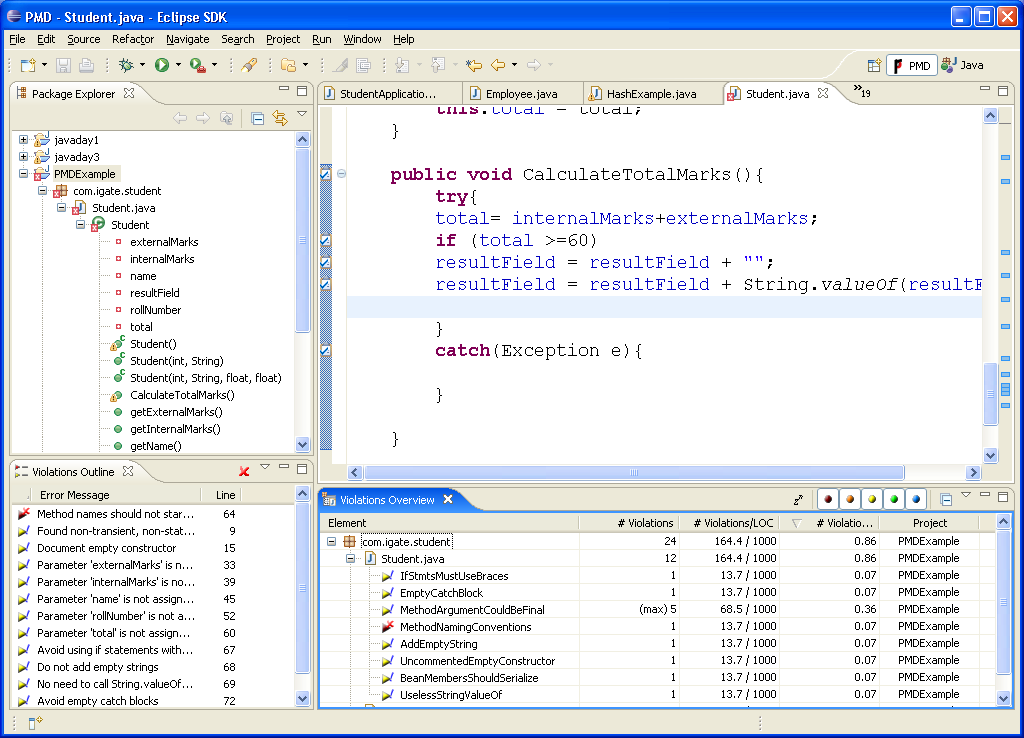


Figure 18: Eclipse showing with Violations

**Step 4: Right click project select PMD – Generate Report**

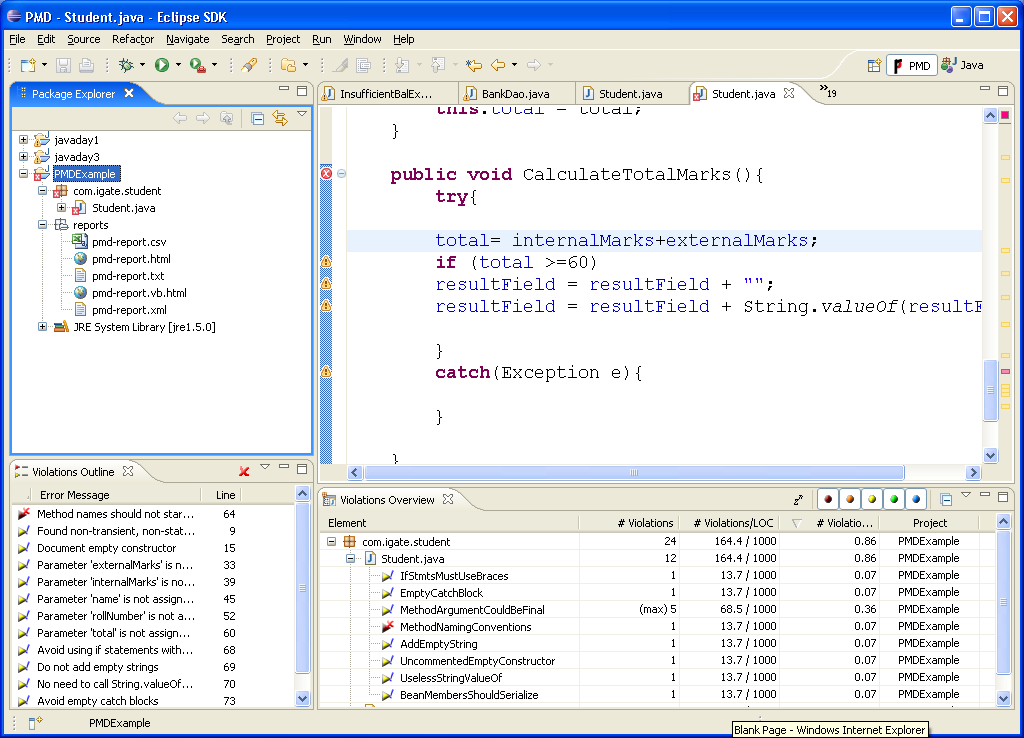
****

Figure 19: Generate Report

**Step 5: Double click pmd-report.html**

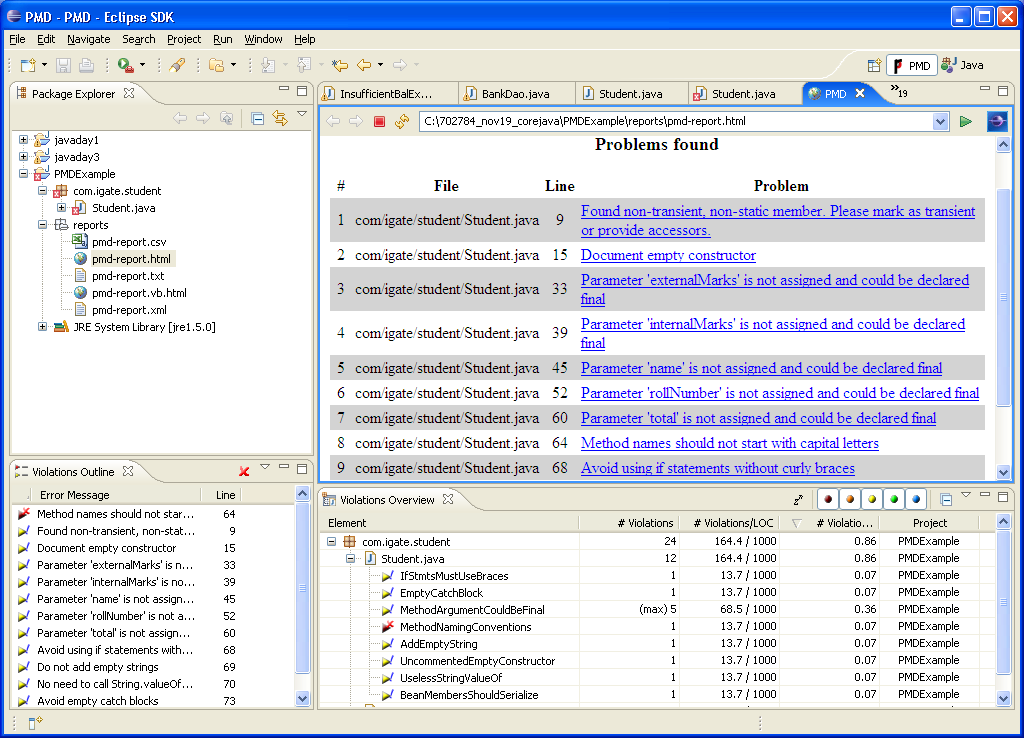
****

Figure 20: HTML Reporting Format

**Step 6: Double click pmd-report.html**

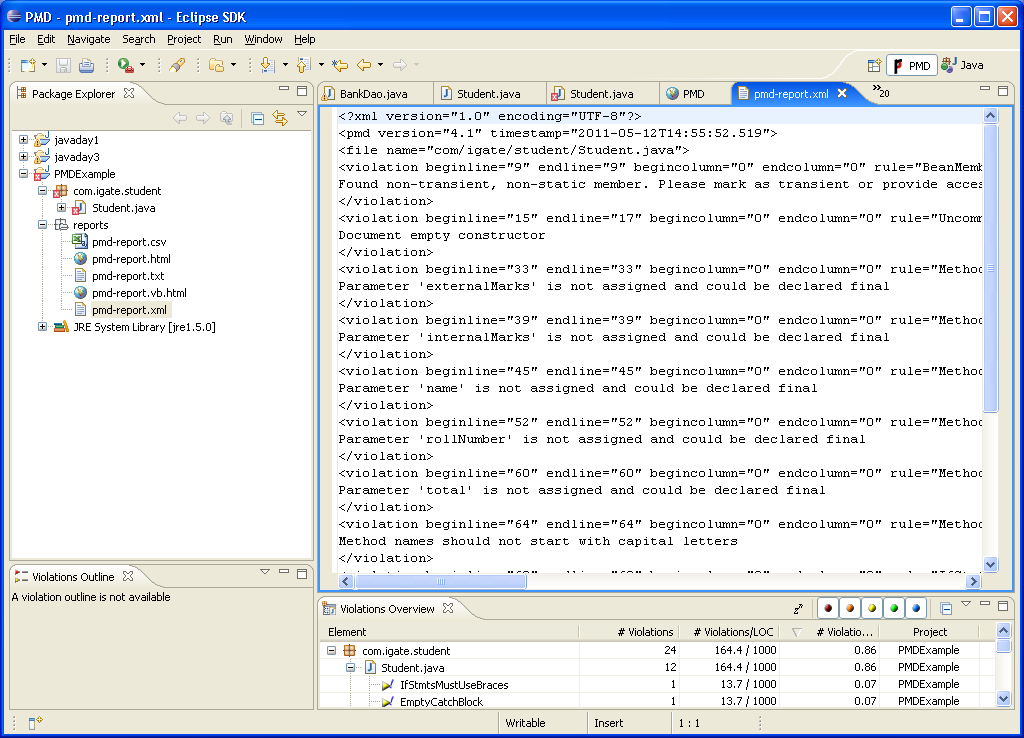


Figure 21: XML Reporting Format

## << TO DO>>

**Assignment-1:**

Review the java code created in the core java Lab Assignment - 4 using PMD tool. Generate the HTML and XML reports.

Appendices

## Appendix A: Table of Figures

[Figure 1: Environmental Variable 5](#_Toc454275706)

[Figure 2: System Properties 6](#_Toc454275707)

[Figure 3: Environment Variables 7](#_Toc454275708)

[Figure 4: Edit User Variable 7](#_Toc454275709)

[Figure 5: Edit User Variable 8](#_Toc454275710)

[Figure 6: Edit User Variable 8](#_Toc454275711)

[Figure 7: settings.xml 9](#_Toc454275712)

[Figure 8: Creating first standalone Maven application using archetype 10](#_Toc454275713)

[Figure 11: Select Wizard in Eclipse 14](#_Toc454275714)

[Figure 12: New Maven Project 15](#_Toc454275715)

[Figure 13: Selecting an archetype 16](#_Toc454275716)

[Figure 14: Specifying Archetype Parameters 17](#_Toc454275717)

[Figure 15: Building an Application 18](#_Toc454275718)

[Figure 16: Enable PMD 19](#_Toc454275719)

[Figure 17: check code with PMD 20](#_Toc454275720)

[Figure 18: Eclipse showing with Violations 20](#_Toc454275721)

[Figure 19: Generate Report 21](#_Toc454275722)

[Figure 20: HTML Reporting Format 21](#_Toc454275723)

[Figure 21: XML Reporting Format 22](#_Toc454275724)