CS335 Software Engineering

Lab 1: Javadocs

Javadoc is a documentation generator from Sun Microsystems for generating documentation in HTML format from Java source code.

This documentation describes the classes but omit all the implementation details, such as the body of all method definitions.

You should comment your classes in a way that you can get the most value out *Javadoc*.

The Java language supports three types of comments:

Comment	Description
/* text */	The compiler ignores everything from /* to */.
// text	The compiler ignores everything from // to the end of the line.
/** documentation */	This is a documentation comment and in general it's called doc comment. The JDK *javadoc* tool uses doc comments when preparing automatically generated documentation.

A doc comment is written in HTML, for Javadoc to extract a comment, the comment must follow these requirements:

i) It immediately precedes a public class definition, a public method definition, or other public items

ii) The comment is in between /** and */, e.g. /** this is my comment */

Other comment styles and comments preceding any private items will not be extracted.

Tags:

A tag is a key word, which allows to describe the characteristic of an item followed it. If a @ tag preceded an item/text, Javadoc will extract that item/text and include it in the documentation with an associated text format.

This is an example of doc comments in the Circle class:

```
import java.util.Scanner;

/**
 * This class performs calculations on circle.
 * It includes 2 methods for calculating area and circumference of a circle
 * @author Author's name
 * @version 1.0
 * @since 08 Feb 2016
 */
public class Circle {

      /**
       * PI attribute
       */
       static double  PI=3.14159;

      /**
       * Calculate area of a circle
       * @param radius double
       * @return area double
       */
      public static double area(double radius)
      {
            return PI*radius*radius;
      }

      /**
       * Calculate circumference of a circle
       * @param radius double
       * @return circumference
       */

      public static double circumference(double radius)
      {
            return PI*2*radius;
      }

      /**
       * Main method asks user to input a radius
       * then displays to console the area and circumference
       * of the corresponding circle
       * @param args
       */
      public static void main(String[] args)
      {

            System.out.println("Enter a radius: ");
            Scanner scanner = new Scanner(System.in);
            double radius = scanner.nextDouble();
            System.out.println("A circle of radius "+ radius + " m has an
area of "+ Circle.area(radius) + " m2");

            System.out.println("A circle of radius "+ radius + " m has a
circumference of "+ Circle.circumference(radius) + " m");
```

```
                scanner.close();
        }
}
```

<u>Exercise 1- Tutorial:</u>

In this tutorial, you will create the Circle class in Eclipse and insert doc comments as above, then generate Javadoc. Perform the steps follows:

a. Open Java Eclipse, create a new Java Project, name it as Exercise1 (File-> New->Java Project

b. Create a package named Shapes (right click on Exercise1 project in the Package Explorer, New->Package)

c. Create a class named Circle (right click on package Shapes, New->Class), then create the 3 methods *area*, *circumference* and *main* as above.

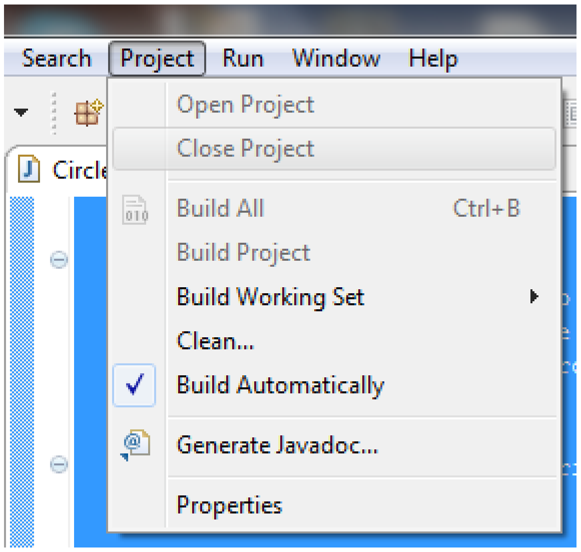
d. Insert doc comments

e. Run the file to see how it works (right click on class Circle, Run as -> Java application)

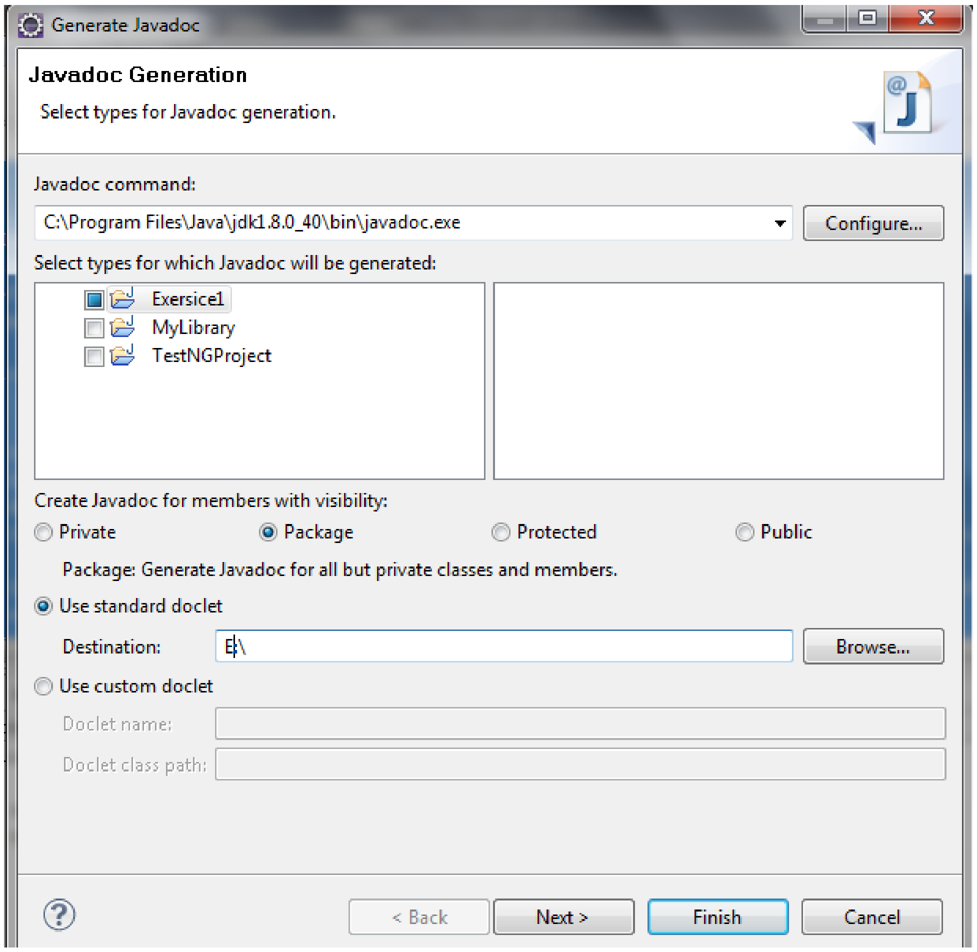
Creating Javadoc:

f. Generate the javadoc as follows:

    i. Select Project Menu, Generate Javadoc…



    ii. The following dialog is displayed:


Generate Javadoc

Javadoc Generation

Select types for Javadoc generation.

Javadoc command:

C:\Program Files\Java\jdk1.8.0_40\bin\javadoc.exe

Select types for which Javadoc will be generated:

- [✓] Exercise1
- [ ] MyLibrary
- [ ] TestNGProject

Create Javadoc for members with visibility:

Private [✓ Package] Protected Public

Package: Generate Javadoc for all but private classes and members.

[✓ Use standard doclet]

Destination: E:\

Use custom doclet

Doclet name:

Doclet class path:

- If the javadoc command is not set – point it to the bin directory in JDK where the file javadoc.exe is. It is something like:

C:\Program Files\Javadk1.6.0_07\bin\javadoc.exe (the java jdk directory on your machine may have a different name)

- Select where you want the documents to be generated by setting the directory for *Use standard doclet* on your X drive.

- Click Finish, then open the document in a web browser to see its content.

Exercise 2 – Calculate on Isosceles angled triangle

a. Now create a class Iso_Angled_Triangle with 2 methods to calculate the area and circumference of an Isosceles angled triangle

b. Insert doc comment to your methods

c. Generate Javadoc

d. Open the documentation in a web browser to see its contents

Exercise 3 – Bank Account

You are provided 2 classes named BankAccount and Transactions. Class BankAccount contains methods for initialize a balance of an account, withdraw and deposit money to the account. Class Transactions allows user to perform transactions on this account.

You need to create a new project, and a package, then import these two classes to your project. Try to compile and run the Java program (Transactions) to see how it works and insert comments to these classes. Finally generate Javadoc for this package.

a. Download the programs BankAccount.java and Transactions.java to your hard drive

b. Create a new project and give it a new name

c. Create a new package in the src folder called atm

d. Import the 2 files BankAccount and Transactions to your atm package (right click on package name, import -> file systems, then select the java files in your hard drive).

e. Run the Transactions program and observe the Java Swing option pane use. This is a simple Swing GUI program. Note the import javax.swing.JOptionPane statement at the top of the program. Note that the program has not updated yet the balance after making withdraw or deposit .

f. Add the suitable code to the Transactioins program where the comments say

   a. //deposit an amount in the account and

   b. //withdraw and amount from the account

   Note: This code should use the object ba and the methods of BankAccount. Pay attention on the return value of each method.

g. Include as many javadoc comments in each method of both BankAccount and Transactions to inform the user of the functionality of the method. Also include javadocs for the overall project to inform the user of who wrote the docs, the version number, etc.

The following document provides more details on different tags in Javadoc.

http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html