**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 its called **doc comment**. The **JDK javadoc** tool uses *doc comments* when preparing automatically generated documentation. |

This tutorial is all about explaining Javadoc. We will see how we can make use of Javadoc for generating useful documentation for our Java code.

Documenting your code is crucial to help others understand it, and even to remind yourself how your own older programs work. Unfortunately, it is easy for most external documentation to become out of date as a program changes. For this reason, it is useful to write documentation as comments in the code itself, where they can be easily updated with other changes. Javadoc is a documentation tool which defines a standard format for such comments, and which can generate HTML files to view the documentation from a web broswer. (As an example, see Oracle's Javadoc documentation for the Java libraries at <http://download.oracle.com/javase/6/docs/api/>.)

Javadoc is a tool which comes with JDK and it is used for generating Java code documentation in HTML format from Java source code which has required documentation in a predefined format.

## **Writing Javadoc Comments**

In general, Javadoc comments are any multi-line comments ("/\*\* ... \*/") that are placed before class, field, or method declarations. They must begin with a slash and two stars, and they can include special tags to describe characteristics like method parameters or return values. The HTML files generated by Javadoc will describe each field and method of a class, using the Javadoc comments in the source code itself. Examples of different Javadoc comments are listed below.

**Simple Comments.**Normal Javadoc comments can be placed before any class, field, or method declaration to describe its intent or characteristics. For example, the following simple Student class has several Javadoc comments.

/\*\*

\* Represents a student enrolled in the school.

\* A student can be enrolled in many courses.

\*/

public class Student {

/\*\*

\* The first and last name of this student.

\*/

private String name;

/\*\*

\* Creates a new Student with the given name.

\* The name should include both first and

\* last name.

\*/

public Student(String name) {

this.name = name;

}

}

**Using Tags.**Tags can be used at the end of each Javadoc comment to provide more structured information about the code being described. For example, most Javadoc comments for methods include "@param" and "@return" tags when applicable, to describe the method's parameters and return value. The "@param" tag should be followed by the parameter's name, and then a description of that parameter. The "@return" tag is followed simply by a description of the return value. Examples of these tags are given below.

/\*\*

\* Gets the first and last name of this Student.

\* @return this Student's name.

\*/

public String getName() {

return name;

}

/\*\*

\* Changes the name of this Student.

\* This may involve a lengthy legal process.

\* @param newName This Student's new name.

\* Should include both first

\* and last name.

\*/

public void setName(String newName) {

name = newName;

}

Some javadoc tags

|  |  |  |
| --- | --- | --- |
| **Tag** | **Description** | **Syntax** |
| @author | Adds the author of a class. | @author name-text |
| @deprecated | Adds a comment indicating that this API should no longer be used. | @deprecated deprecated-text |
| @exception | Adds a **Throws** subheading to the generated documentation, with the class-name and description text. | @exception class-name description |
| @param | Adds a parameter with the specified parameter-name followed by the specified description to the "Parameters" section. | @param parameter-name description |
| @return | Adds a "Returns" section with the description text. | @return description |
| @throws | The @throws and @exception tags are synonyms. | @throws class-name description |
| @version | Adds a "Version" subheading with the specified version-text to the generated docs when the -version option is used. | @version version-text |

## Example:

Following program uses few of the important tags available for documentation comments. You can make use of other tags based on your requirements.

The documentation about the AddNum class will be produced in HTML file AddNum.html but same time a master file with a name index.html will also be created.

import java.io.\*;

/\*\*

\* <h1>Add Two Numbers!</h1>

\* The AddNum program implements an application that

\* simply adds two given integer numbers and Prints

\* the output on the screen.

\* <p>

\* <b>Note:</b> Giving proper comments in your program makes it more

\* user friendly and it is assumed as a high quality code.

\*

\* @author Zara Ali

\* @version 1.0

\* @since 2014-03-31

\*/

public class AddNum {

/\*\*

\* This method is used to add two integers. This is

\* a the simplest form of a class method, just to

\* show the usage of various javadoc Tags.

\* @param numA This is the first paramter to addNum method

\* @param numB This is the second parameter to addNum method

\* @return int This returns sum of numA and numB.

\*/

public int addNum(int numA, int numB) {

return numA + numB;

}

/\*\*

\* This is the main method which makes use of addNum method.

\* @param args Unused.

\* @return Nothing.

\*/

public static void main(String args[]) throws IOException

{

AddNum obj = new AddNum();

int sum = obj.addNum(10, 20);

System.out.println("Sum of 10 and 20 is :" + sum);

}

}

Now, process above AddNum.java file using javadoc utility as follows:

C:\> javadoc AddNum.java

This generates the the corresponding Javadoc HTML files

You can view the Javadoc files by opening them in a browser

Extracted from:

<http://www.tutorialspoint.com/java/java_documentation.htm>

http://www.drjava.org/docs/user/ch10.html