



Comments in Java are used to add explanatory notes to the code. These notes **are not executed** by the compiler and are therefore not part of the program's functionality. Instead, they serve as notes that help developers understand the code and make it more readable.

In simple words, Comments are non-executable code that is used to document the code.

There are three types of comments in Java



1

Single line comment

2

Multi line comment

3

Javadoc comment or Documentation comment

Single line Comments in Java

Single-line comments start with **two forward slashes (//)** and continue to the end of the line. Anything after the slashes is considered a comment and is ignored by the compiler. Here are the few examples:


```
int age; // Represents the age of the customer

// Maximum number of login retry attempts allowed
static final int MAX_RETRY_ATTEMPTS = 3;

// y indicates YES
char y = 'Y';

// n indicates NO
char n = 'N';
```

Using single line comment in the middle of the java statement is not valid as it fails the compilation of the java statement,



```
int num // Sample value= 0;
```

Single line comments are also used for commenting the Java statements instead of removing them completely. This will give flexibility for the developers to retain the code for later use or testing

```
public static void main(String[] args) {
    // Comment out the following line for alternative behavior
    // performAlternativeOperation();
}
```

Multi line Comments in Java

Multi-line comments starts with a forward slash immediately followed by an asterisk (/*) and ends with an asterisk immediately followed by a forward slash (*). Anything between these two characters is considered a comment and is ignored by the compiler. It can be used to explain a complex code snippet, documenting a class, method or to comment multiple lines of code at a time. Below are examples,

```
/*
  This class represents a customer in Bank app.
  It contains information such as firstName,
  lastName, DOB, address and account details.
*/
public class Customer {
}
```

```
/*
  This method calculates the area of the rectangle based
  on given length and width. The logic it is going to have
  is multiply two given input method arguments and return
  the same to the caller
*/
public double calculateRectangleArea (double length, double width) {
    return length*width;
}
```

```
/*
  Based on business requirements, the end user is
  allowed to retry a maximum of 3 login attempts.
  Using the below constant, the same is going to be
  controlled. If the requirement changes, then
  change the below value
*/
static final int MAX_RETRY_ATTEMPTS = 3;
```

Multi line Comments in Java

Using multi line comment, we can comment a piece of code like a entire method, a block of code, entire class etc.

```
/*public int sum (int num1, int num2) {  
    return num1 + num2;  
}*/
```

Single line comment can also be used to mention multiple line of comments or comment multiple lines of code. But using multi line comment approach is more of convinient,

```
/*  
    This is a multi-line comment.  
    It can span more than one line.  
*/  
  
// This is a multi-line comment.  
// It can span more than one line
```

A multiline comment can be inserted in the middle of Java code, but it is not recommended to use this way. The compiler ignores all text starting from `/*` to `*/`:

```
static final int MAX_RETRY_ATTEMPTS /* Comment */ = 3;
```

Javadoc comments are a special type of comment used in Java programming language to document code. Javadoc comments start with **/** (two asterisks) and end with */ (an asterisk and a slash)**. The JDK provides a **javadoc** command-line tool to extract the documentation comments from the source code and generate documentation in HTML format.

Although a documentation comment can appear anywhere in Java source code, the javadoc tool uses only those documentation comments to generate the HTML pages that appear just before the declaration of classes, nested classes, interfaces, enums, annotations, constructors, methods, and fields.

```
/**
 * Takes two {@code int} numbers as input
 * and add them.
 * <p>
 *     <b> For example, 2 + 3 =5 </b>
 * @param num1 Represent first number
 * @param num2 Represent second number
 * @return sum value of first and second number
 */
public int sum (int num1, int num2) {
    return num1 + num1;
}
```

In this example, the Javadoc comment starts with a brief description of what the method does. The @param tag is used to document each parameter, and the @return tag is used to document the return value. HTML elements like <p>, , <i> </i> etc. can also be used in Javadoc comments.

Javadoc comments are important for documenting code and making it easier to understand and maintain. They also allow other developers to use your code more easily by providing documentation that explains how to use it.

Java Doc comment tags

Below are the most commonly used tags in Javadoc comments,

Tag	Description	Applicable to
@since	Conveys under which version the item was added	Variable
@version	Indicates Version number	Class
@author	Author name	Class
@see	Related class name	Class, method, variable
@link	Related URL	Class, method, variable
@code	Source code content	Class, method, variable
@param	Method param name & description	Method
@deprecated	Marks an item obsolete	Class, method, variable
@return	Details of the return value	Method
@exception	Details of the exception	Method