Java If Else

Java If Else

Java If Else statement can be used to implement decision making logic in your Java applications.

You can use if else statement in situations where you need to select one of the two code blocks for execution based on a logical condition.

We shall start with simple If statement, where there would be no else block. Then we shall discuss about if-else, and later on we shall look into if-else-if ladder.

In this tutorial, we will learn the syntax and examples for if statement, if-else statement and if-else-if statement.

Syntax

If Statement

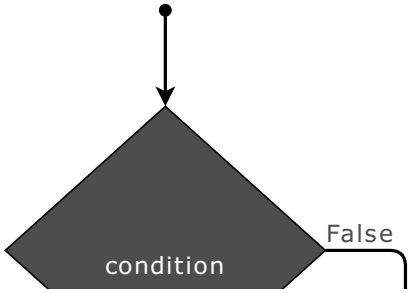
Following is the syntax of simple if statement in Java.

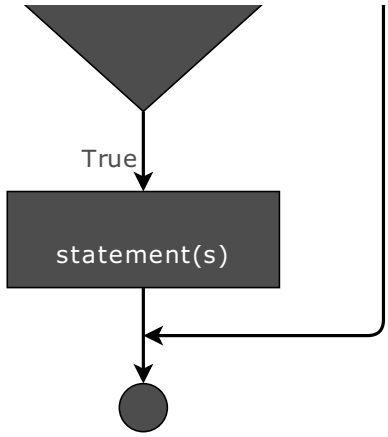
```
if(condition) {
     //if code block
 }
```

The condition is an expression that should evaluate to a boolean value.

If the `condition` evaluates to `true`, if code block is executed. Else, continue with the subsequent statements.

Following is the flow diagram of if statement.





If-Else Statement

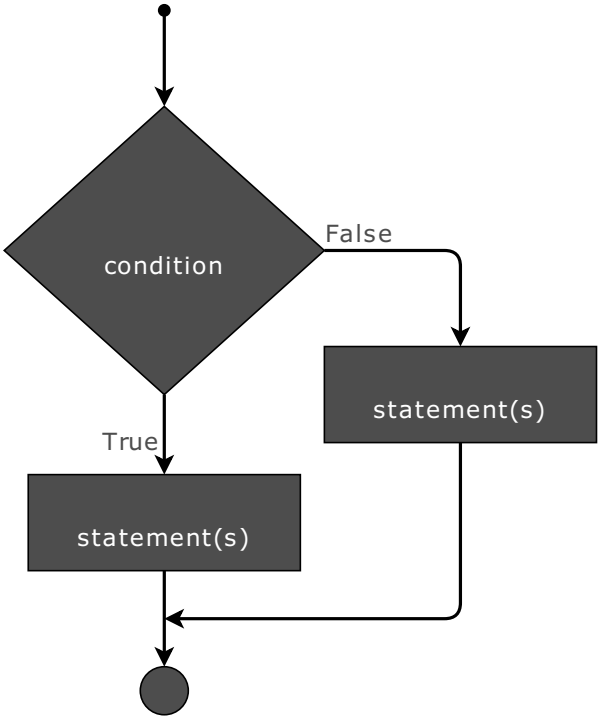
Following is the syntax of if else statement in Java.

```
if(condition) {
     //if code block
 } else {
     //else code block
 }
```

If the `condition` evaluates to `true`, if code block is executed.

If the `condition` evaluates to `false`, else code block is executed.

Following is the flow diagram of if-else statement.



If-Else-If Statement

Following is the syntax of if else statement in Java.

```
if(condition1) {
     //statement(s)
 } else if(condition2) {
     //statement(s)
 } else if(condition3) {
     //statement(s)
 } else {
     //statement(s)
 }
```

If the `condition1` evaluates to `true`, execute corresponding block of statements and the execution of if-else-if completed. If the `condition1` evaluates to `false`, check `condtition2`.

Now, If the `condition2` evaluates to `true`, execute corresponding block of statements and the execution of if-else-if completed. If the `condition2` evaluates to `false`, check `condtition3`.

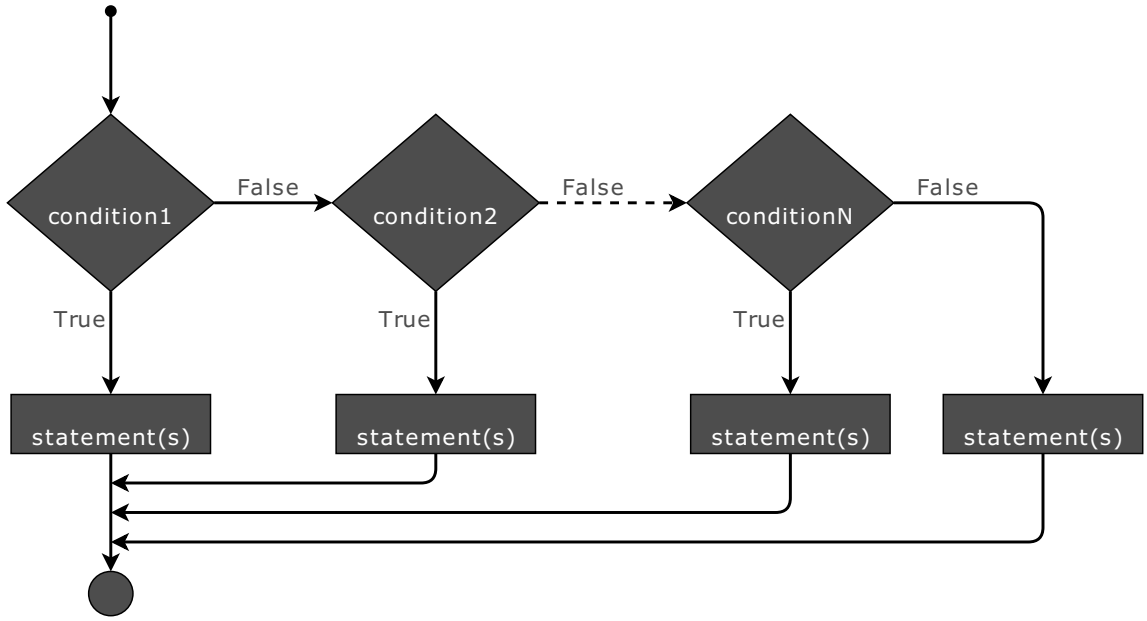
If the `condition3` evaluates to `true`, execute corresponding block of statements and the execution of if-else-if completed. If the `condition1` evaluates to `false`, execute else block.

Please note that there could be as many number of **else if** blocks as required.

else block at the end of if-else-if ladder is optional.

To put in simple terms, from top to bottom, the first condition that gets evaluated to true executes the corresponding block of statements.

Following is the flow diagram of if-else-if statement.



Example 1 – If statement

In this example, we shall write a simple if statement to check if variable a has a value of 3 or not.

Java Program

```
/**
 * Java Program - IF statement example
 */

public class IfExample {

    public static void main(String[] args) {
        int a = 3;

        if(a==3) {
            System.out.println("a is 3.");
        }
    }
}
```

Run the program, and you shall see that as the condition `a==3` evaluates to **true**, if block is executed.

Output

```
a is 3.
```

Now, initialize `a` with a value other than `3` and run the program. As the condition `a==3` evaluates to **false**, if block will not execute.

Example 2 – If Else Statement

In the following example, we have an if else statement. In the condition, we are checking if `a` is `2`.

Java Program

```
/**
 * Java Program - IF ELSE statement example
 */

public class IfElseExample {

    public static void main(String[] args) {
        int a = 3;

        if(a == 2) {
            System.out.println("a is 2.");
        } else {
            System.out.println("a is not 2.");
        }
```

```
    }
}
```

As the condition is **false** with `a=3`, else block will execute.

Run the above Java Program, and you shall get the following output.

Output

```
a is not 2.
```

If the condition were true, if block would execute.

Example 3 – Nested If-Else Statement

You can write an if else statement inside another if else statement. This is called nesting, and the corresponding if-else statement is called nested if-else statement.

In the following example, we have written an if-else statement, inside the if block.

```
/**
 * Java Program - IF-ELSE statement example
 */

public class NestedIfElseExample {

    public static void main(String[] args) {
        int a = 3;

        if(a % 2 == 1) {
            System.out.println("a is odd number.");
            if(a<10) {
                System.out.println("a is less than 10.");
            } else {
                System.out.println("a is not less than 10.");
            }
        } else {
            System.out.println("a is even number.");
        }
    }
}
```

Run the program and you will get the following output in console.

Output

```
a is odd number.
a is less than 10.
```

Example 4 – If-Else-If Statement

In the following example, we shall write an if-else-if statement.

In the following example, we shall check if a number is divisible by 2, 3 or 5.

Java Program

```
/**
 * Java Program - IF-ELSE-IF statement example
 */

public class IfElseIfExample {

    public static void main(String[] args) {
        int a = 15;

        if(a % 2 == 0) {
            System.out.println("a is divisible by 2.");
        } else if(a % 3 == 0) {
            System.out.println("a is divisible by 3.");
        } else if(a % 5 == 0) {
            System.out.println("a is divisible by 5.");
        } else {
            System.out.println("a is not divisible by 2, 3 or 4.");
        }
    }
}
```

a%2==0 is false. So, check the next condition. a%3==0 is true. Execute the corresponding block. If-else-if statement execution is done.

When you run the above Java Program, you shall get the following output.

Output

```
a is divisible by 3.
```

Conclusion

In this Java Tutorial, we learned how to write if statement, if-else statement, nested if-else statement and if-else-if statement in Java programs.

- Java Tutorial
- Java Introduction
- Java Installation
- IDEs for Java Development
- Java HelloWorld Program
- Java Program Structure
- Java Datatypes
- Java Variable Types
- Java Access Modifiers
- Java Operators
- Java Decision Making
- Java Loops
- Java Array
- Java OOPs
- Java String
- Java Exception Handling
- Java File Operations
- Java Date & Time
- Java MySQL
- Java Random
- Java Math