Java For Loop

Java For Loop

Java For Loop can be used to execute a set of statements in a definite or indefinite loop based on a condition.

Java For loop functionality is same as that of Java While loop, except that the initialization and update can be defined in for loop definition iteself.

In this tutorial, we will learn how to define/write a Java For loop, and its usage using example programs.

Syntax of Java For Loop

The syntax of Java For loop is given below.

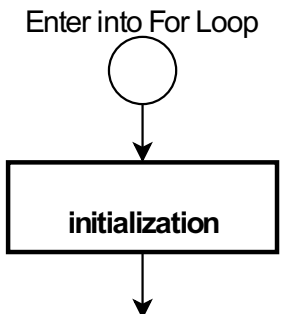
```
for(initialization; condition; update) {
     //statements
 }
```

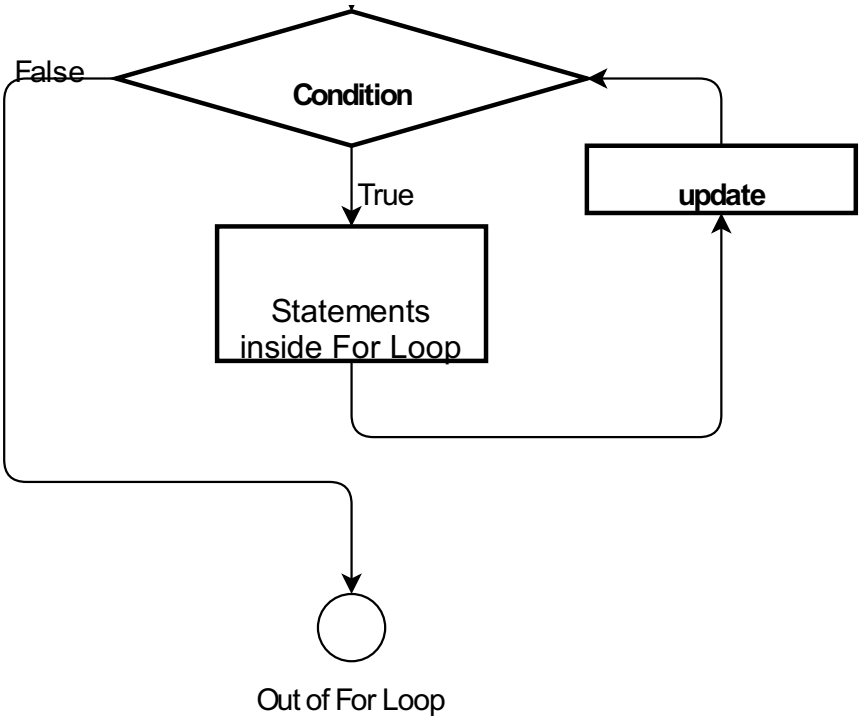
where

- **initialization** can contain one or more variables defined and initialized.
- **condition** is the expression which is checked during each iteration of the loop.
- **update** can contain increment, decrement, or any update to the variables. Variables defined in the **initialization** part can be updated here.

Working of Java For Loop

Following flowchart depicts the working of Java For loop.





`initialization` part is executed only once. That too before even checking the condition in its first run.

The execution stays in the loop until the **condition** evaluates to `true`.

During each iteration, after the statements inside for loop are executed, **update** section gets executed.

If the **condition** evaluates to `false`, the control comes out of the For loop and continues with the execution of statement after for loop statement.

Example 1 – Java For Loop

Following is a simple example to implement Java For loop. In the initialization, we have defined a variable `i` and assigned a value `0` to it. During each iteration we check if i is less than `5` and increment `i` at the end of each iteration.

Example.java

```
public class Example {

    public static void main(String[] args) {
        for(int i=0;i<5;i++) {
            System.out.println(i);
        }
    }

}
```

When you run the above program, the following is printed to the console.

Output

```
0
1
2
3
4
```

Example 2 – Java For Loop – Array

In this example, we will use for loop to traverse through all the elements of an array. We start with index, `i=0` and continue till `i` is less than the length of array, by incrementing `i` at the end of each loop iteration.

Example.java

```
public class Example {

    public static void main(String[] args) {
        String[] names = {"Apple", "Google", "Apache"};

        for(int i=0;i<names.length;i++) {
            System.out.println(names[i]);
        }
    }

}
```

When you run the above program, all the elements of the array are printed to the console as shown in the following.

Output

```
Apple
Google
Apache
```

Example 3 – Java Enhanced For Loop

Java Enhanced For Loop iterates just like the normal for loop, but it eliminates redundancy of initializing a variable for index, increment it over each iteration and accessing the element of iterable. Please observe the for loop in the following example.

Example.java

```
public class Example {

    public static void main(String[] args) {
```

```
        String[] names = {"Apple", "Google", "Apache"};

        for(String name: names) {
            System.out.println(name);
        }
    }

}
```

When you run the above program, the following is printed to the console.

Output

```
Apple
Google
Apache
```

Example 4 – Java For Loop – Indefinite Loop

Following is a simple example to implement for loop to iterate indefinitely. Condition part of for loop is optional. And we are not checking any condition in the for loop. So, the for loop runs indefinitely.

Example.java

```
public class Example {

    public static void main(String[] args) {
        for(int i=0; ;i++) {
            System.out.println(i);
        }
    }

}
```

When you run the above program, the natural numbers are printed indefinitely.

This example is just for demonstration of what happens if your condition does not break the loop ever.

These situations should actually be avoided when you are developing an application, unless required specifically as per client requirement.

Example 5 – Java Nested For Loop

You can write a for loop inside a for loop.

Following is a simple example where we print tower of stars using nested for loop.

Example.java

```
public class JavaTutorial {

    public static void main(String[] args) {
        for(int i=1;i<6;i++) {
            for(int j=0;j<i;j++) {
                System.out.print("*");
            }
            System.out.println();
        }
    }

}
```

When you run the above program, the following pattern is printed to the console.

Output

```
*
**
***
****
*****
```

Example 6 – Java For Loop – Break

For loop generally breaks when the condition is evaluated to `false`. But, you can also break the loop from within the body of for loop using break statement.

In this example, we shall break the for loop when i becomes 4.

Example.java

```
public class Example {

    public static void main(String[] args) {
        for(int i=1;i<10;i++) {
            if(i==4) {
                break;
            }
            System.out.println(i);
        }
    }

}
```

When you run the above program, the following is printed to the console.

Output

```
1
2
3
```

Example 7 – Java For Loop – Continue

You can skip the execution of loop or part of the loop for that cycle using continue statement.

In this example, we shall skip the execution of for loop body when i equals 4.

Example.java

```
public class Example {

    public static void main(String[] args) {
        for(int i=1;i<8;i++) {
            if(i==4) {
                continue;
            }
            System.out.println(i);
        }
    }

}
```

When you run the above program, the following is printed to the console with the execution of for loop skipped at `i=4`.

Output

```
1
2
3
5
6
7
```

Conclusion

In this Java Tutorial, we learned how to write a For loop in a Java program and different scenarios where for loop can be used.

Java Tutorial