Huy Nguyen

ENG 2140 - Technical Description First Draft

9/16/24

**Introduction**

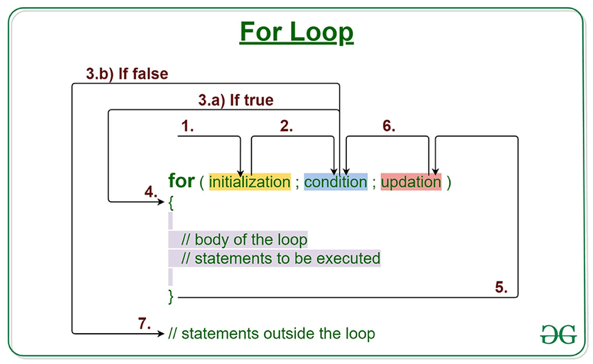
In Java and almost every other programming language, the technique of executing a piece of code multiple times in a loop is extremely important. This technique helps reduce redundant lines of code when performing calculations or printing operations that require repetition in programming. There are four typical types: for loop, while loop, do-while loop, and for-each loop. The biggest difference between these four loops is that the for loop is used when the number of iterations is fixed, while while and do-while loops are more appropriate when the number of iterations is not fixed. In addition, the for-each loop allows the user to check list and see what the list contains for the user to access the list one at a time.

**For Loop**

In Java, a for loop is used to repeat a piece of code a certain number of times or iterate over the elements of an array or a list. The for loop contains the initialization and increment/decrement conditions in one line, so the structure of the loop is shorter and less error-prone. Figure 1 describes how this method works:

* Initialization Expression: First, the loop begins with a variable declared by the user.
* Condition: A test and comparison expression that returns a Boolean value. The loop is executed when the return value is true and exits when it is false.
* Update Expression: This updates the value of the variable for the next iteration if the loop continues.

After completing one iteration of the loop, the condition is re-evaluated. The process continues until the condition is false.



Figure

**While Loop**

A while loop is a flow control statement that allows code to be executed multiple times based on a certain Boolean condition. The while loop can also be thought of as a repeating if statement. In addition, the while loop operates almost like a for loop, except the while loop allows the user to have more control of the loop. Figure 2 shows:

* The while loop starts by checking the condition. If the condition is true, the statements in the loop will be executed. If the condition is false, the loop is skipped.
* Statements containing updated values for variables are processed for the next iteration.
* When the condition becomes false, the loop will end.

A diagram of a company

Description automatically generated

Figure

**Do-While Loop**

The do-while loop has a similar function to the while loop, but the main difference is that do-while checks the condition after executing the statements. Therefore, the do-while loop executes the loop contents at least once, even if the condition is false. Figure 3 shows:

* The do-while loop starts executing statements without checking any conditions first.
* After executing the statements and updating variable values, the condition is checked.
* If the condition is true, the next iteration starts. If the condition is false, the loop ends.

The do-while loop will always execute the statements at least once before any condition is tested.

A diagram of a diagram

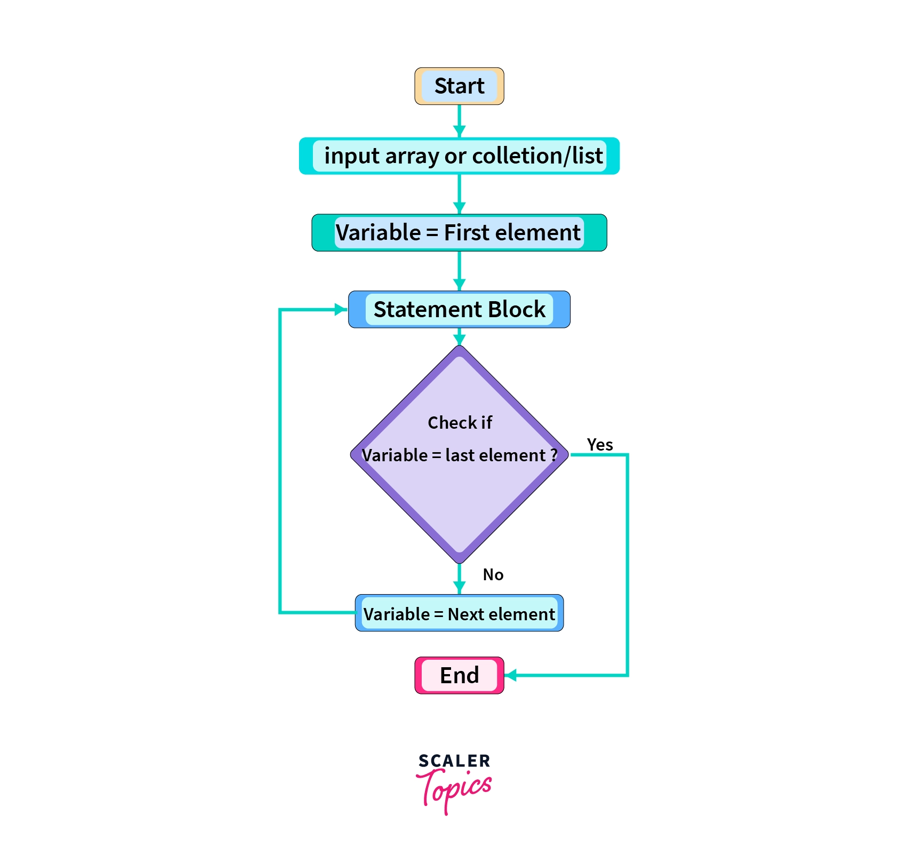
Description automatically generated

Figure

**For-each Loop**

In Java, there are classes that extend to interface, called the iterator class. For those classes, users may implement something called a for-each loop that allow the user to see and check whatever it is in the list. The user must declare a local variable with a specific type that the list contains. For example, if the list contains integer, users may not use different wrapper or primitive types. Once the type and local variable are in part of the condition, users may use it to iterate the elements inside of the list without using abundant lines of codes. As shown in figure four, it will check the list and store in a local variable. Then, it will constantly check if it is the end of the list. Once it goes through the list, the loop will end. Some properties that for-each loop has:

* For-each loops introduce aliases. This means that the list is a reference type. Users access the list, using reference to see what is inside the list. Now, introducing it means that there are multiple sources that will lead to the same reference, which might lead to problems.
* If the type, the user declared, happens to be a mutable type (type that can be changed in reference type), the for-each will not work because the local variable will not change with the loop body.



Figure

**Conclusion**

Loops in Java are a flexible structure that helps programmers automate repetitive tasks. By understanding different types of loops, such as for, while, or do-while, and how to use them effectively, programmers can write concise and optimized JavaScript code easily. With loops, efficiency of the codes enhances the usability and work coherently with other parts of the code to produce an effective solution to the tasks.