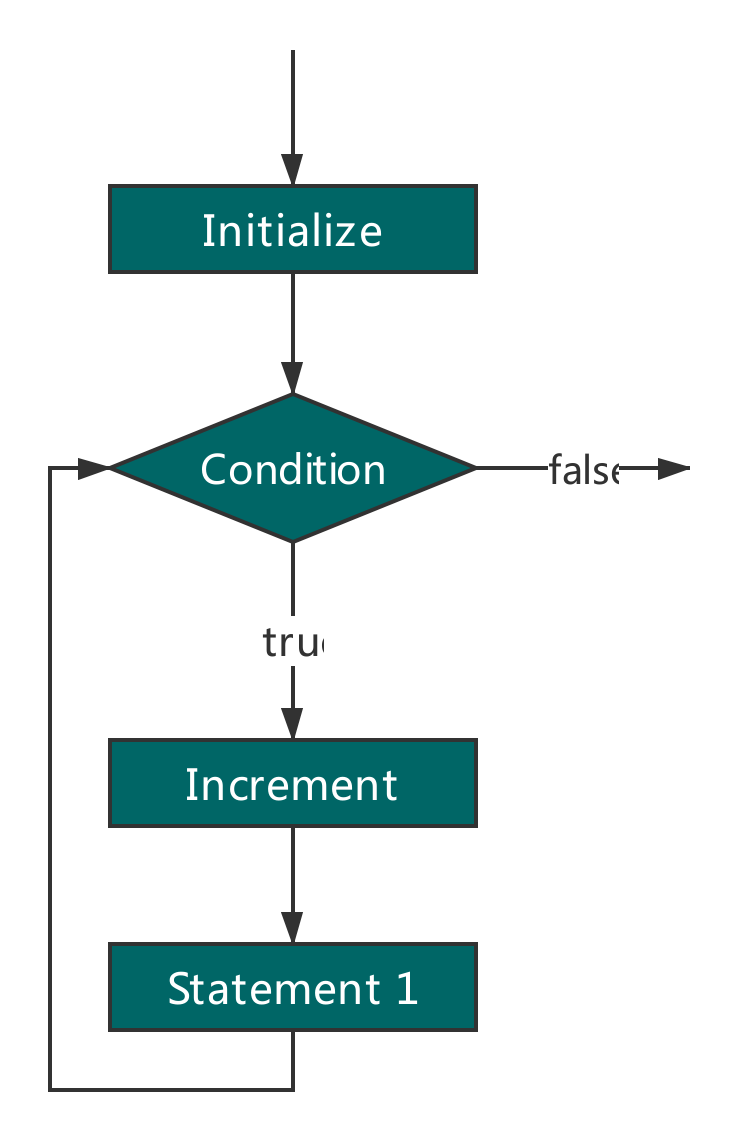
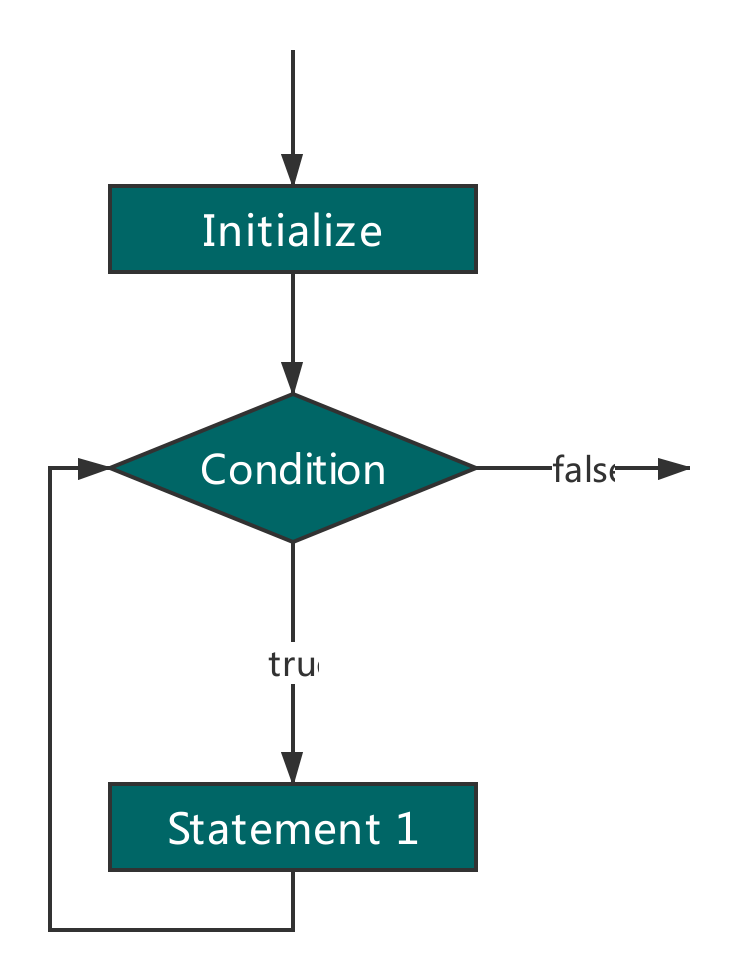
**Learning Objectives**

* To grasp loop implementation methods in JavaScript.

**Learning Contents**

* In the preceding part, we learned **if else** and **switch**. Branch structure enables JavaScript to make judgment, but the computer’s judgment ability is far different from human beings’. Computer is good at another thing, i.e. repetition or loop. In a program design, loop is a method to repeatedly execute an identical code stock, which may reduce workload of repeated writing of source program. In a program design, loop is very useful and also a program structure which can most effectively exert strengths of computers. For example: if we want to execute the same operation on every item of an array, a loop structure may be used.
* Three elements of a loop: **loop variable**、**loop body** and **loop termination condition**. Loop variable is a variable that indicates the current loop in the loop structure; loop body is the program to be repeatedly executed in the loop structure; and loop termination condition is normally a conditional judgment statement, which is used to judge whether current loop structure is to be terminated otherwise it will be an infinite loop. If the condition is satisfied, codes of the loop body is executed; if the condition is not satisfied, the execution will jump out of the loop and the codes behind the loop structure will be executed.
* In JavaScript, there are normally three methods with which we can implement a loop structure:
  + For loop
  + Flow diagram of the **for** statement is shown as below:
* 
  + In this loop structure, when the condition is satisfied, statement 1 will be executed; otherwise the execution will jump out of the loop body.
  + Basic syntax of the for statement is shown as below:
  + **for (loop variable initialization; loop termination condition; increment;){**
  + **loop body;**
  + **}**
  + Next, let’s see a simple case: there are ten students counting off, “Student No. 1、Student No. 2……”. With for loop, we can use just a few codes to realize counting of ten students.
  + **for(var i = 1; i<=10; i++){**
  + **console.log('Student No. ' + i);**
  + **}**
  + Output is shown as below:
  + Student **No. 1**
  + **Student No. 2**
  + **Student No. 3**
  + **Student No. 4**
  + **Student No. 5**
  + **Student No. 6**
  + **Student No. 7**
  + **Student No. 8**
  + **Student No. 9**
  + **Student No. 10**
  + In this case, the loop is executed for 10 times. So does the 10 in **for(i=1; i<=10; i++)** means 10 times? Let’s look at working mechanism of **for** loop at first. First of all, **i=1** is called an initial condition, which indicates the starting point. In our case, it starts from **i=1**. **i<=10** appearing behind the first semicolon indicates a loop termination condition. Each loop will first judge whether this condition is satisfied. If the condition is satisfied, the loop will go on; otherwise the loop will stop and continue to execute codes behind the **for** loop. You might ask, if we set **i=0**, it will be less than or equivalent to 10 forever? Let’s look at the third part. The final **i++** means **i** plus **1**. This is the action after every loop. That is to say, when the loop ends at every time, **i** will be **1** greater than original value. After execution of many loops, the condition of **i<=10** is not satisfied any more. Then the loop ends. Codes behind the for loop will be executed.
* While loop
  + **while** loop executes a piece of code repeatedly until certain condition is not satisfied any more.
  + Flow diagram of the **while** statement is shown as below:
* 
  + **while** statement is a pre-test loop statement. In other words, before execution of bodes in the loop body, the value of exported condition will be got. Therefore, codes in the loop body might never be executed. The syntax of **while** statement is shown as below:
  + **while(loop termination condition){**
  + **loop body**
  + **}**
  + Being similar with **for** loop, when the loop condition is executed, the statement within the braces **{}** will be executed; otherwise, the execution will jump out of the loop. Now let’s implement the above case with **while** loop:
  + **var i = 1;**
  + **while(i<=10){**
  + **console.log('Student No. ' + i );**
  + **i = i + 1;**
  + **}**
  + Implementation results of these two cases are basically the same except for the implementation form. It seems there is something missing in **while** loop in comparison with the **for** loop. It has only one judgment condition. However, **while** loop also has initial conditions, which have been defined before, such as **var i = 1;** in the above case. Increment of the variable i is placed within the loop body. Actually, this process has no difference with the **for** loop, with the variable i increasing gradually until non-satisfaction of the judgment condition and loop termination.
  + do-while loop
  + do-while loop statement is a post-test loop statement. Only when codes within the loop body are executed, the exported conditions will be tested. In other words, before the value of the conditional expression is got, codes within the loop body will be executed once at least. The syntax of the do-while statement is shown as below:
  + **do{**
  + **loop body**
  + **}while(loop termination condition)**
  + We can also realize the above case with the following method:
  + **var i = 1;**
  + **do{**
  + **console.log('Student No. ' + i);**
  + **i = i + 1;**
  + **} while(i<=10)**
  + Output of this case is the same as the above result. We can see that do-while is also similar to while. It also makes initialization outside the loop and changes loop variables within the loop body. The difference lies in whether loop termination condition is judged in advance or not.
  + Such post-test loop statements as do-while which are mostly used in loop body are executed once at least.

**Recommended Resources**

* JS Loop - MDN(<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Loops_and_iteration>)
* How to Build Loop Structure in JS(<https://www.digitalocean.com/community/tutorials/how-to-construct-for-loops-in-javascript>)
* JS Loop(<http://www.dofactory.com/tutorial/javascript-loops>)