NAZARBAYEV UNIVERSITY

SCHOOL OF ENGINEERING AND DIGITAL SCIENCES

Web Programming and Problem Solving

JavaScript Loops

Date: 26.10.2022

Instructor: Zhandos Yessenbayev

Content

- Problems
- For-loop
- While-loop
- More loops

Problem

Problem 1. **Too many steps to repeat!**

In the Lab8, you were asked to compute the sums for the each lab, say like:

```
let sum1 = lab1[0]+lab1[1]+lab1[2]+lab1[3]+lab1[4]+lab1[5]+lab1[6]
```

What if you were asked to **sum 1000 elements** in the array?!

NAZARBAYEV UNIVERSITY

Problem

<u>Problem 2.</u> Repetition based on condition

“Don’t practice until you get it right.

$$\underbrace{\textbf{Practice}}_{\text{Repetitive task}}\ \underbrace{\textbf{until you can’t get it wrong.”}}_{\text{Condition}}$$

Loops

To perform some **repetitive** task, we use **loops**.

Loops are JavaScript constructs that allow us to perform the repetitive tasks:

- ***for*** a specified number of times;
- ***while*** a specified condition holds true.

For-Loop

Syntax

```
for (expression 1; expression 2; expression 3) {
  // code block to be executed
}
```

where:

- **Expression 1** is executed (one time) before the execution of the code block.
- **Expression 2** defines the condition for executing the code block.
- **Expression 3** is executed (every time) after the code block has been executed.

For-Loop

```
let lab1 = [5,5,0,5,5,10,10];
let sum1 = 0;
for (let i = 0; i < lab1.length; i++) {
  sum1 += lab1[i];
}
```

Comments

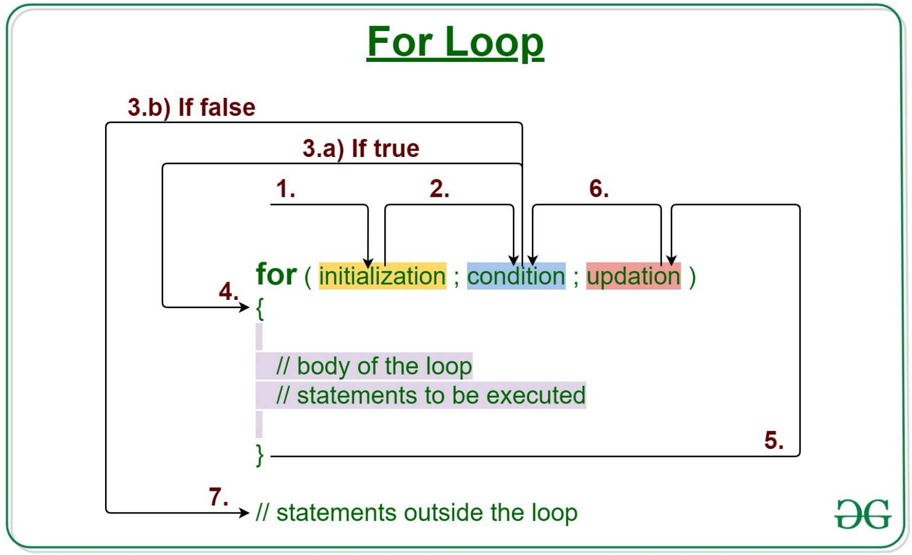
i – loop variable

Three steps of the loop:
1) let i = 0 – initialization
2) i < lab1.length – stop condition
3) i++ - update rule

NAZARBAYEV UNIVERSITY

For-Loop

```
let lab1 = [5,5,0,5,5,10,10];
let sum1 = 0;
for (let i = 0; i < lab1.length; i++) {
  sum1 += lab1[i];
}
```


For Loop
3.b) If false
3.a) If true
1.
2.
6.
for (initialization ; condition ; updation)
4.
{
// body of the loop
// statements to be executed
}
5.
7.
// statements outside the loop

i	*i < lab1.length*	*Condition*	lab1[i]	*sum1*
0	0<7	true	5	5
1	1<7	true	5	10
2	2<7	true	0	10
3	3<7	true	5	15
4	4<7	true	5	20
5	5<7	true	10	30
6	6<7	true	10	40
7	7<7	false	-	-

For-Loop

What does these (nested) loops do?

```
for (let i = 0; i < 5; i++) {
  for (let j = 0; j < 10; j++) {
    console.log(“Hello, World!”)
  }
}
```

While-Loop

Syntax

```
while (condition) {
  // code block to be executed
}
```

where:

- **Condition** is a logical expression for executing the code block.

While-Loop

```
let lab1 = [5,5,0,5,5,10,10];

let sum1 = 0;

let i = 0;                      // initialization of loop variable

while (lab1[i] > 0) {           // stop condition
   sum1 += lab1[i];             // body of the loop

   i++;                         // update the loop variable
}
```

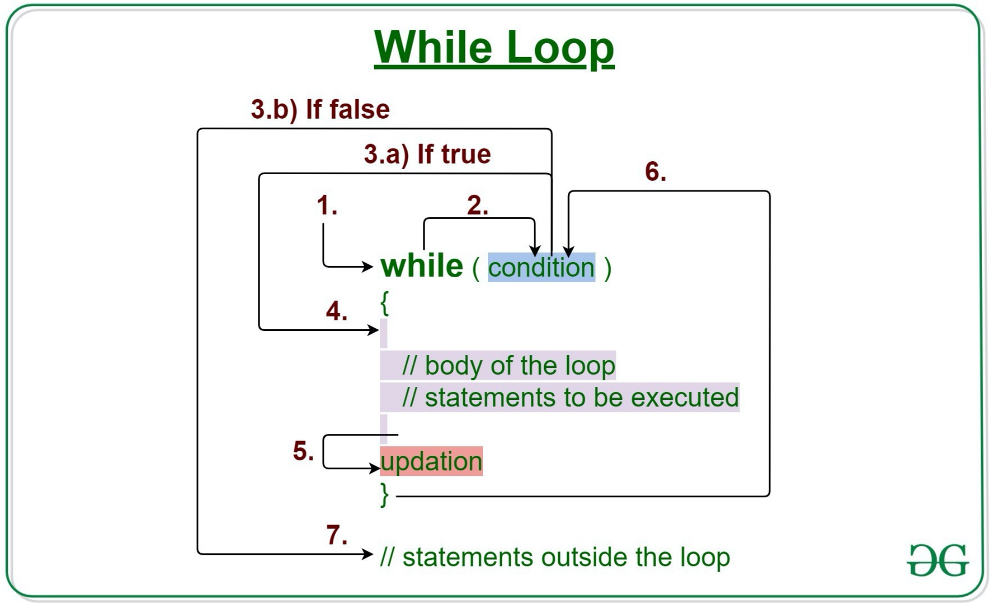
Comments

There are also three steps of the loop:

1) `let i = 0` – initialization
2) `lab1[i] > 0` – stop condition
3) `i++` - update rule

While-Loop

```
let lab1 = [5,5,0,5,5,10,10];

let sum1 = 0;

let i = 0;

while (lab1[i]>0) {
  sum1 += lab1[i];

  i++;         // update

}
```



i	lab1[i]	lab1[i] > 0	*Condition*	*sum1*
0	5	5>0	true	5
1	5	5>0	true	10
2	0	0>0	false	-

While-Loop

What does this loop do?

```
let i = 0;
while (i < 100) {
  console.log(“Hello, World!”)
  i++;
}
```

How does it compare to this loop?

```
for (let i = 0; i < 100; i++) {
  console.log(“Hello, World!”)
}
```

While vs For

When to use while-loop and for-loop?

As a **rule of thumb**:

- Use for-loop when you know in advance the number of steps to do
- Otherwise use while-loop

NAZARBAYEV UNIVERSITY

Break and Continue

There are two useful commands used in loops:

- The break statement exits a loop.
- The continue statement skips one iteration in the loop.

```
for (let i = 0; i < 10; i++) {
  if (i == 4) {
      continue;
  }
  if (i == 8) {
      break;
  }
  console.log(“Hello, World!”)
}
```

What is the output?

More Loops

There other variants of the loops:

- for in statement loops through the properties of an **Object**
- for of statement loops through the values of an iterable object (Array, String)
- forEach() method calls a function once for each **array** element.
- do while loop is a variant of the while loop.

Summary

- **Key takeaways**:
  - To perform repetitive tasks, use loops
  - There are two types of loops that run:
    - for a specified number of times
    - while a specified conditions holds.
  - Don't forget about three steps of the loops:
    - Initialize the loop variables before the loop
    - Setup the condition to exit the loop
    - Update the value of the loop variables.
  - Use break and continue commands if needed

Thanks for Attention!