13. JavaScript For Loop

Loops can execute a block of code a number of times.

13.1 JavaScript Loops

Loops are handy, if you want to run the same code over and over again, each time with a different value.

Often this is the case when working with arrays:

Instead of writing:

```
document.write(cars[0] + "<br>");
document.write(cars[1] + "<br>");
document.write(cars[2] + "<br>");
document.write(cars[3] + "<br>");
document.write(cars[4] + "<br>");
document.write(cars[5] + "<br>");
```

You can write:

```
for (var i=0;i<cars.length;i++)
{
document.write(cars[i] + "<br>");
}
```

13.2. Different Kinds of Loops

JavaScript supports different kinds of loops:

- **for** - loops through a block of code a number of times
- **for/in** - loops through the properties of an object
- **while** - loops through a block of code while a specified condition is true
- **do/while** - also loops through a block of code while a specified condition is true

13.3. The For Loop

The for loop is often the tool you will use when you want to create a loop.

The for loop has the following syntax:

```
for (statement 1; statement 2; statement 3)
  {
  the code block to be executed
  }
```

Statement 1 is executed before the loop (the code block) starts.

Statement 2 defines the condition for running the loop (the code block).

Statement 3 is executed each time after the loop (the code block) has been executed.

Example

```
for (var i=0; i<5; i++)
  {
  x=x + "The number is " + i + "<br>";
  }
```

From the example above, you can read:

Statement 1 sets a variable before the loop starts (var i=0).

Statement 2 defines the condition for the loop to run (i must be less than 5).

Statement 3 increases a value (i++) each time the code block in the loop has been executed.

a) Statement 1

Normally you will use statement 1 to initiate the variable used in the loop (var i=0).

This is not always the case, JavaScript doesn't care, and statement 1 is optional.

You can initiate any (or many) values in statement 1:

Example:

```
for (var i=0,len=cars.length; i<len; i++)
{
document.write(cars[i] + "<br>");
}
```

And you can omit statement 1 (like when your values are set before the loop starts):

Example:

```
var i=2,len=cars.length;
for (; i<len; i++)
{
document.write(cars[i] + "<br>");
}
```

b) Statement 2

Often statement 2 is used to evaluate the condition of the initial variable.

This is not always the case, JavaScript doesn't care, and statement 2 is optional.

If statement 2 returns true, the loop will start over again, if it returns false, the loop will end.

💡 If you omit statement 2, you must provide a **break** inside the loop. Otherwise the loop will never end. This will crash your browser. Read about breaks in a later chapter of this tutorial.

c) Statement 3

Often statement 3 increases the initial variable.

This is not always the case, JavaScript doesn't care, and statement 3 is optional.

Statement 3 could do anything. The increment could be negative (i--), or larger (i=i+15).

Statement 3 can also be omitted (like when you have corresponding code inside the loop):

Example:

```
var i=0,len=cars.length;
for (; i<len; )
{
document.write(cars[i] + "<br>");
i++;
}
```

13.4. The For/In Loop

The JavaScript for/in statement loops through the properties of an object:

Example

```
var person={fname:"John",lname:"Doe",age:25};

for (x in person)
  {
  txt=txt + person[x];
  }
```

You will learn more about the for / in loop in the chapter about JavaScript objects.