Programming of Web Pages

Lecture 4 - JavaScript

What is JavaScript?

Scripting language used on websites

It was created in 1995 by **Netscape**

It supports **functional and object-oriented** programming

It works **on the client side**

Possibilities of JavaScript

Modification of content displayed on the website

Modification of attribute values of HTML tags

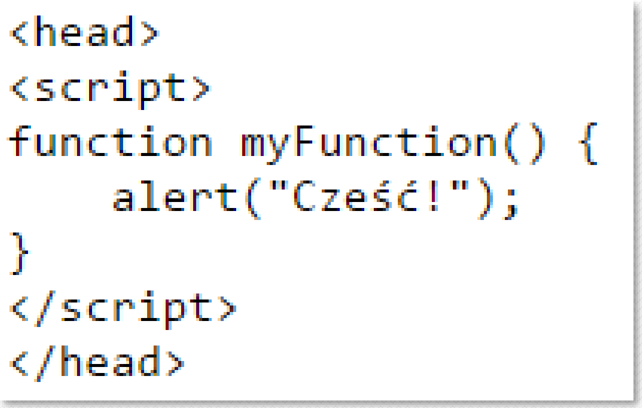
Modification of styles of HTML tags

Preliminary data validation on browser-side

Where to place the script?

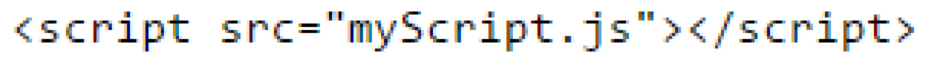
In HTML document script code have to be placed inside tags **<script> … </script>**

Scripts can be places in section **<head>** or **<body>**

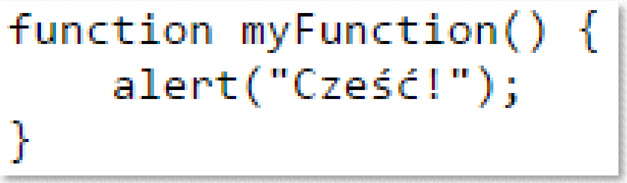


```
<head>
<script>
function myFunction() {
    alert("Cześć!");
}
</script>
</head>
```

Scripts can also be placed in **an external file**



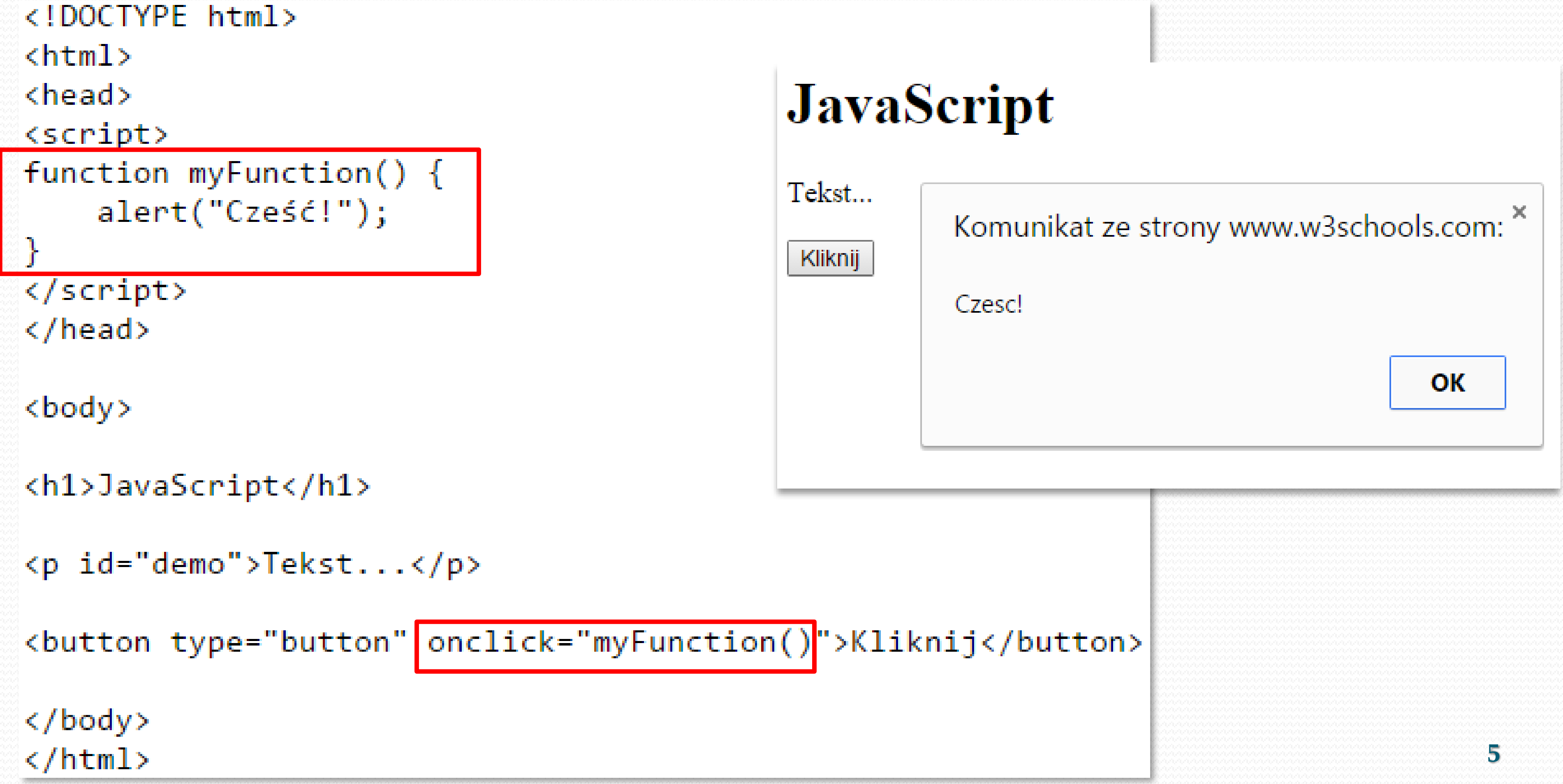
```
<script src="myScript.js"></script>
```



```
function myFunction() {
    alert("Cześć!");
}
```

How to call script?

The scripts are usually in form of a **function**, which is called by the occurrence of an **event**



```
<!DOCTYPE html>
<html>
<head>
<script>
function myFunction() {
    alert("Cześć!");
}
</script>
</head>

<body>

<h1>JavaScript</h1>

<p id="demo">Tekst...</p>

<button type="button" onclick="myFunction()">Kliknij</button>

</body>
</html>
```

JavaScript

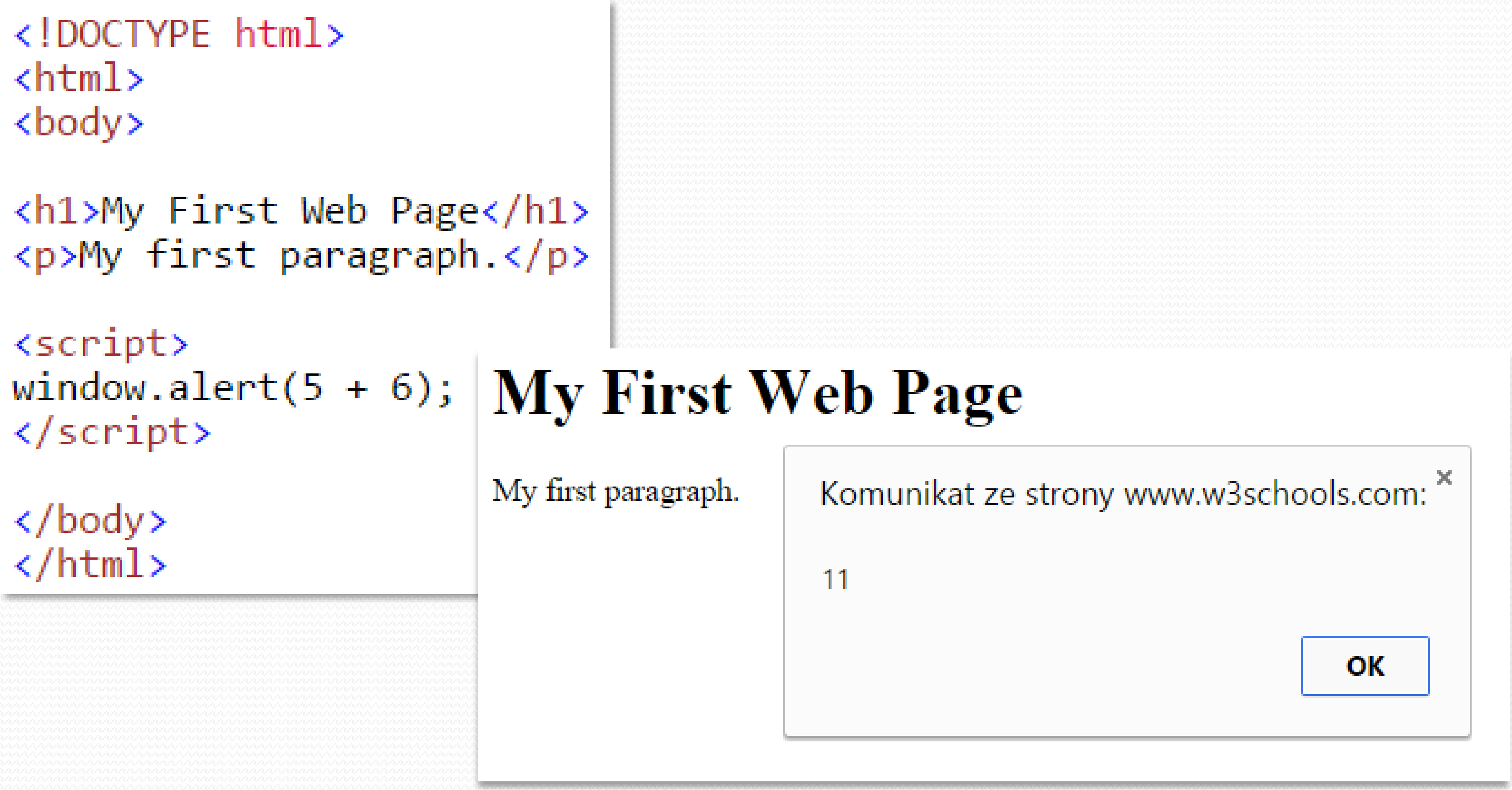
Tekst...

Komunikat ze strony www.w3schools.com:

Czesc!

Output

alert("text") – in the message window



```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
window.alert(5 + 6);
</script>

</body>
</html>
```

My First Web Page

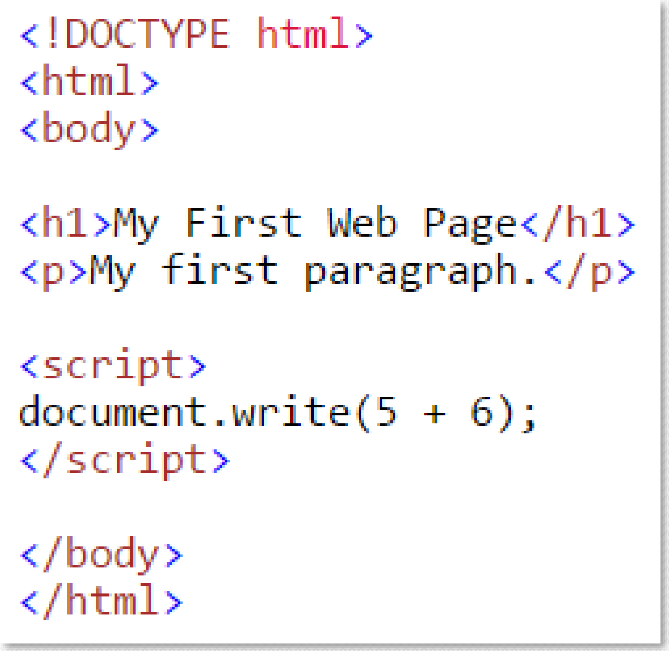
My first paragraph.

Komunikat ze strony www.w3schools.com:

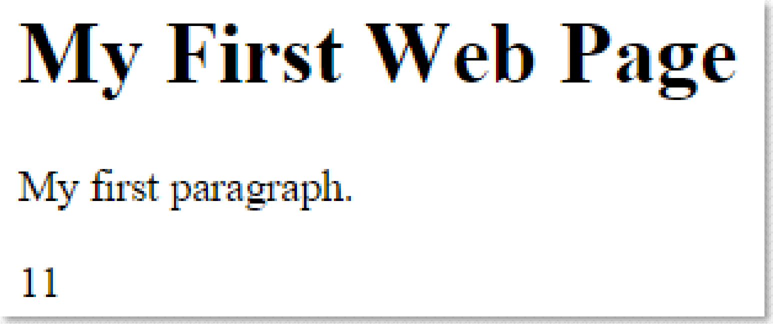
11

Output

document.write("text") – in the document



```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
document.write(5 + 6);
</script>

</body>
</html>
```

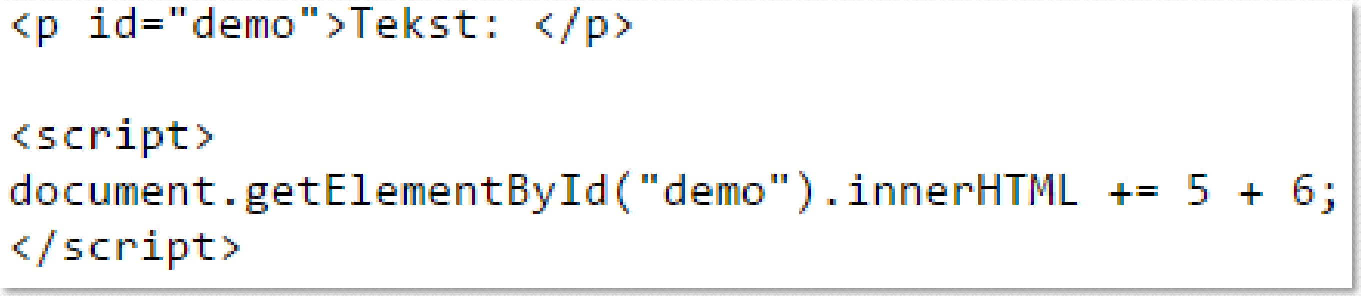


```
My First Web Page

My first paragraph.

11
```

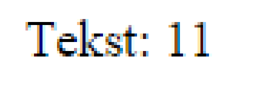
Calling this method by event
clears the entire document (!)

Output

document.getElementById(„tagId").innerHTML – changes the content of the selected tag



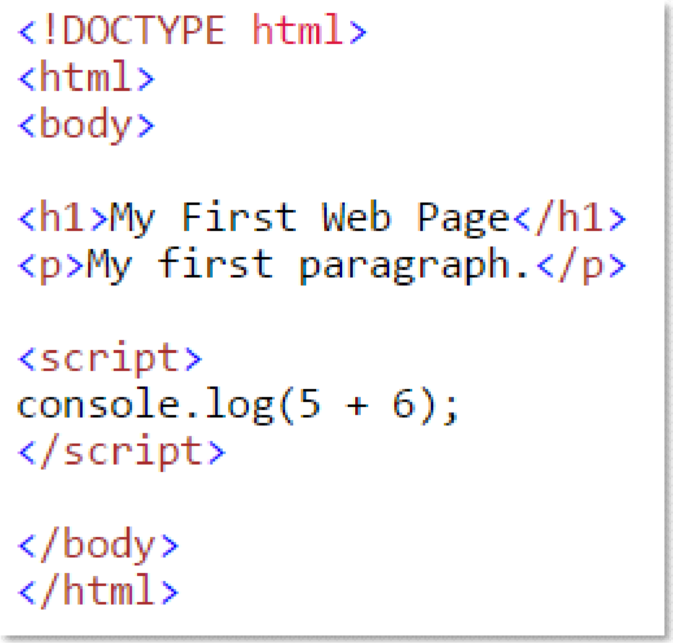
```
<p id="demo">Tekst: </p>

<script>
document.getElementById("demo").innerHTML += 5 + 6;
</script>
```



```
Tekst: 11
```

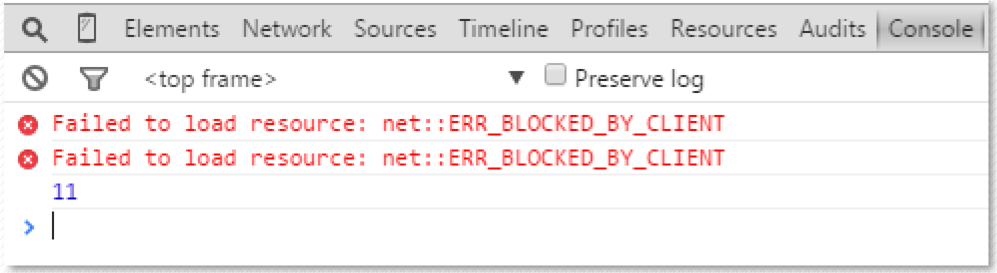
Output

console.log("text") – in the console of the browser



```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
console.log(5 + 6);
</script>

</body>
</html>
```

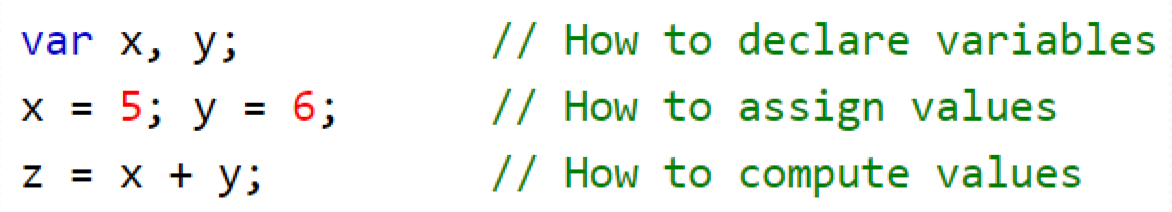
To **preview console** in the browser please press **F12**



```
Failed to load resource: net::ERR_BLOCKED_BY_CLIENT
Failed to load resource: net::ERR_BLOCKED_BY_CLIENT
11
> |
```

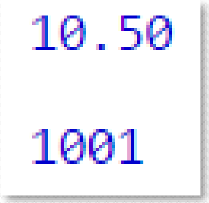
Syntax

At the end of the line we put a **semicolon**



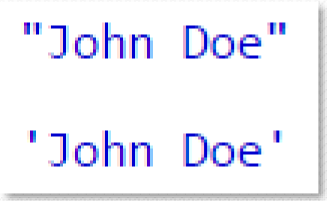
```
var x, y;              // How to declare variables
x = 5; y = 6;          // How to assign values
z = x + y;             // How to compute values
```

Numbers are written with or without decimals



```
10.50

1001
```

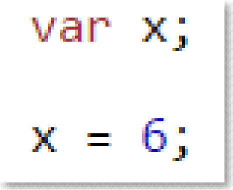
Strings are text, written within double or single quotes:



```
"John Doe"

'John Doe'
```

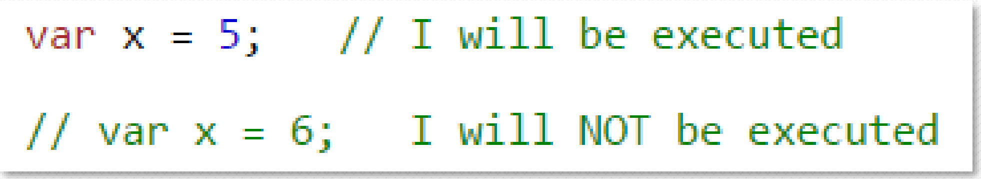
Syntax

In a programming language,
variables are used to store data values.

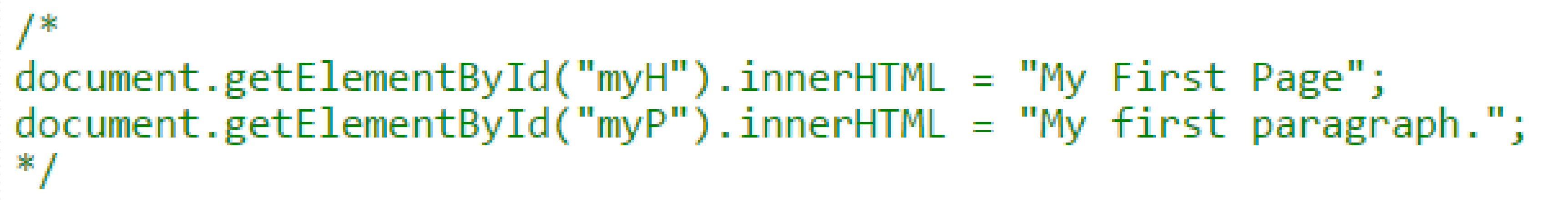


```
var x;

x = 6;
```

Comments



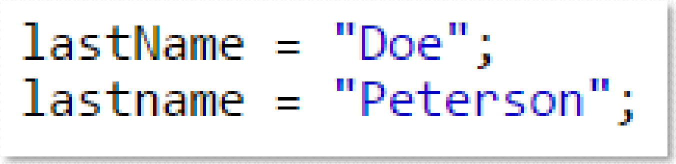
```
var x = 5;    // I will be executed

// var x = 6;   I will NOT be executed
```



```
/*
document.getElementById("myH").innerHTML = "My First Page";
document.getElementById("myP").innerHTML = "My first paragraph.";
*/
```

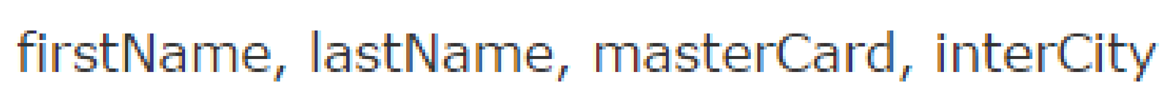
Syntax

Character size



```
lastName = "Doe";
lastname = "Peterson";
```

The naming convention of variables - Camel notation



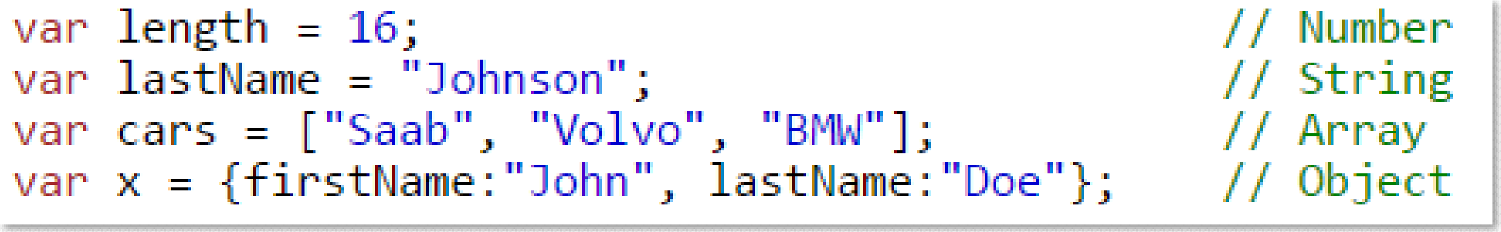
firstName, lastName, masterCard, interCity

Character encoding - the Unicode (UTF-8)

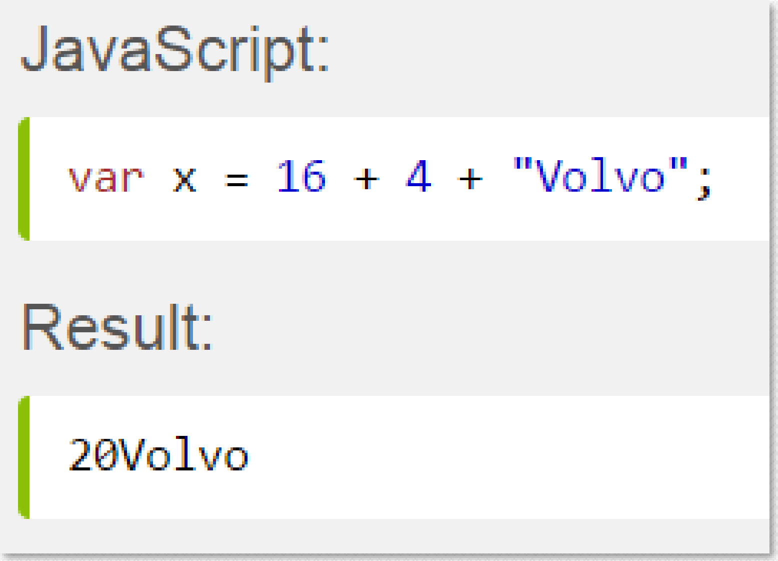
Keywords

Word	Meaning
break	It breaks working of the loop
continue	It ends working of the current loop step
do … while	Loop construction
for	Loop construction
function	Function declaration
if … else	Conditional construction
return	It ends working of the function, it may return a value
switch	Conditional construction
try … catch	Exception handling block
var	Declaration of the variable

Dynamic types



```
var length = 16;                                  // Number
var lastName = "Johnson";                         // String
var cars = ["Saab", "Volvo", "BMW"];              // Array
var x = {firstName:"John", lastName:"Doe"};       // Object
```

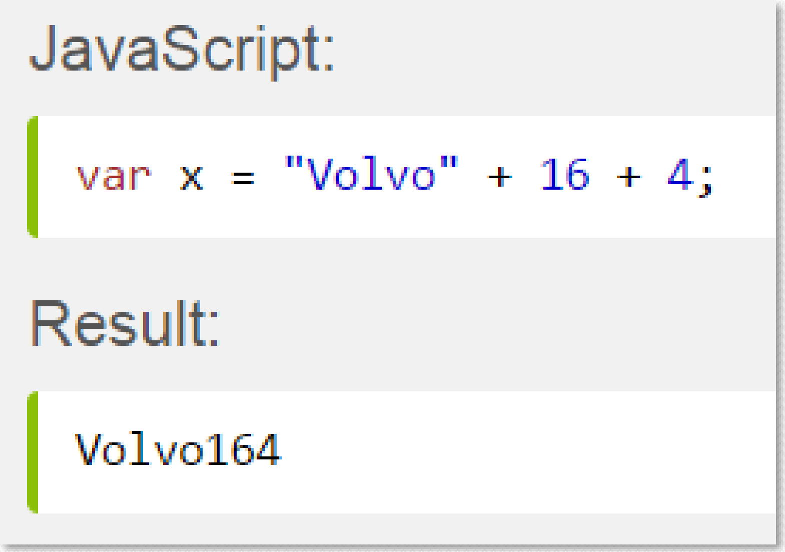


JavaScript:

```
var x = 16 + 4 + "Volvo";
```

Result:

```
20Volvo
```



JavaScript:

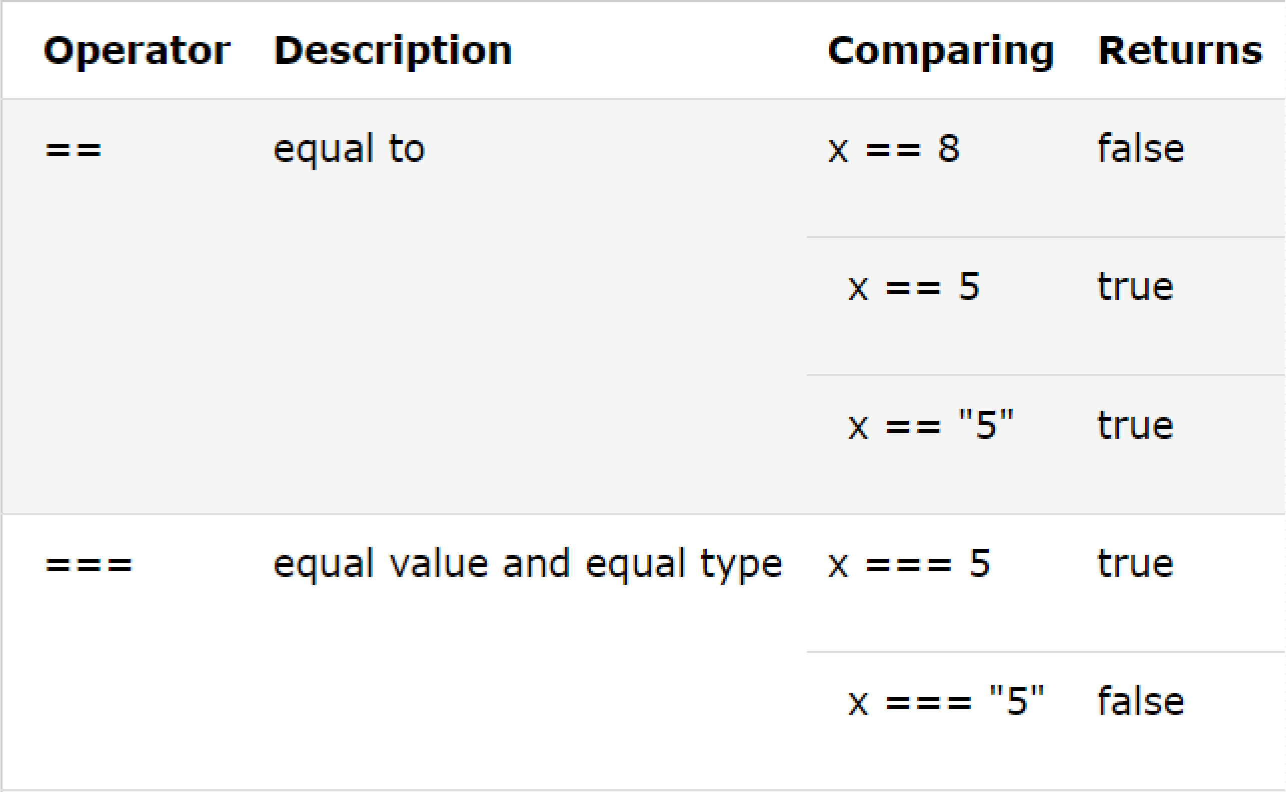
```
var x = "Volvo" + 16 + 4;
```

Result:

```
Volvo164
```

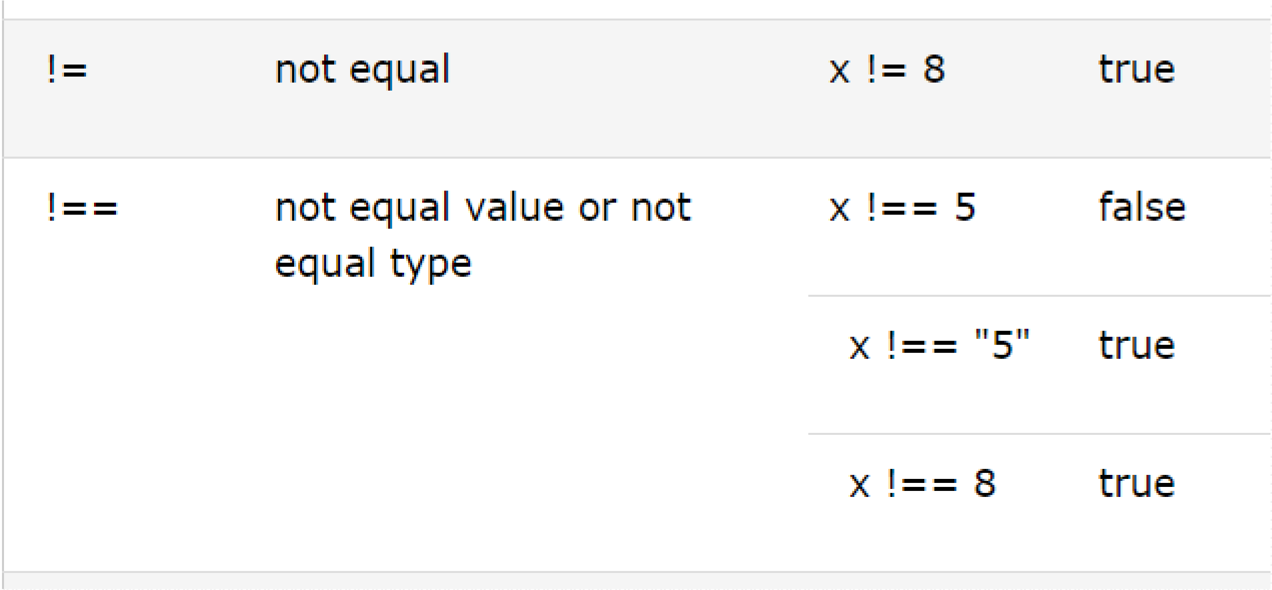
Comparison operators

var x = 5;



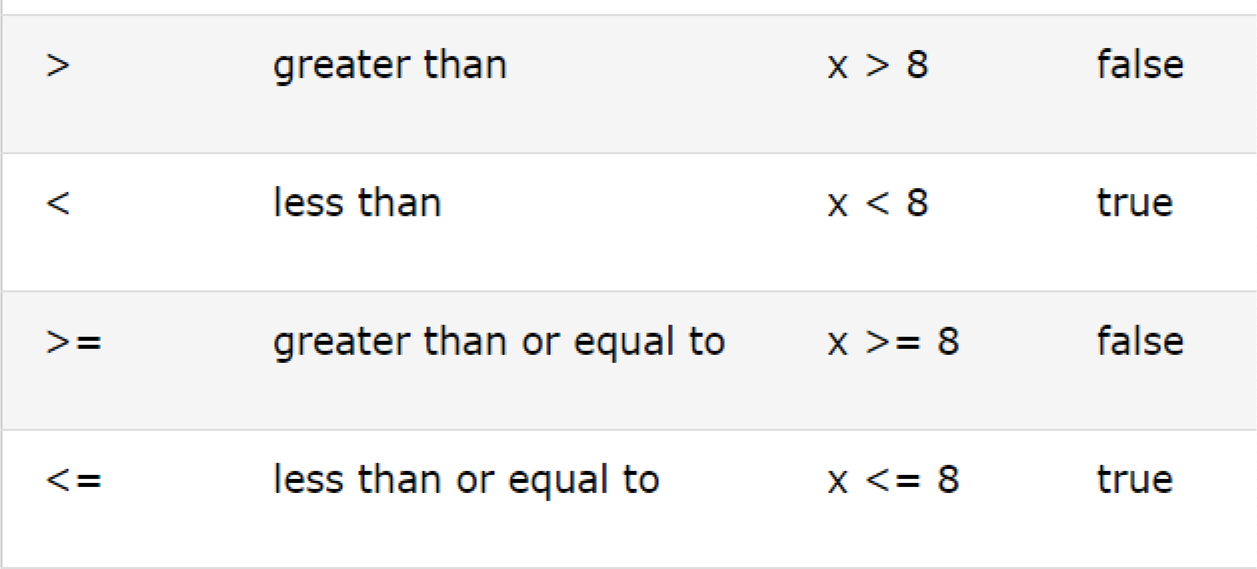
Operator	Description	Comparing	Returns
==	equal to	x == 8	false
		x == 5	true
		x == "5"	true
===	equal value and equal type	x === 5	true
		x === "5"	false

Comparison operators



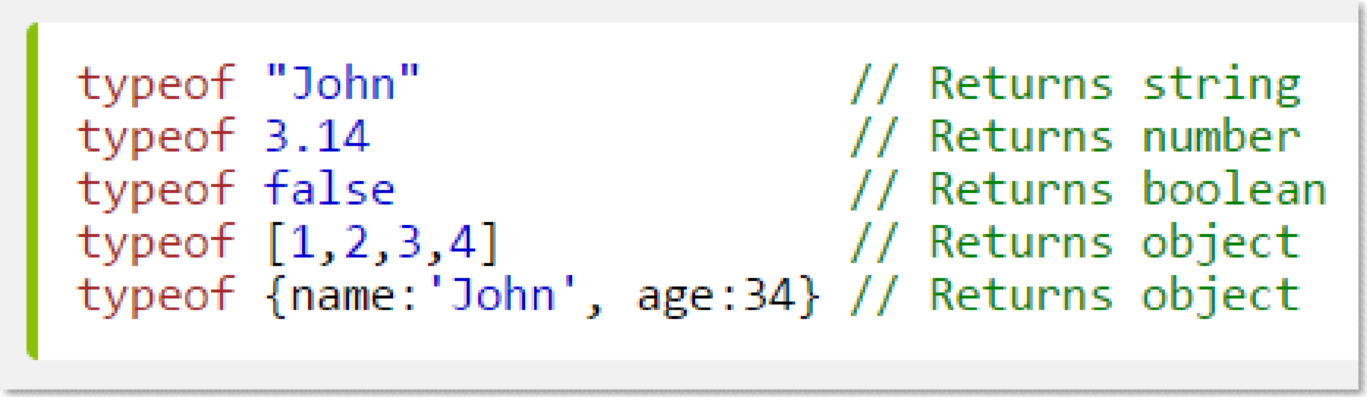
!=	not equal	x != 8	true
!==	not equal value or not equal type	x !== 5	false
		x !== "5"	true
		x !== 8	true

Comparison operators

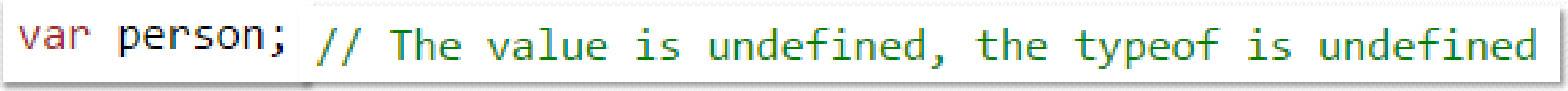


>	greater than	x > 8	false
<	less than	x < 8	true
>=	greater than or equal to	x >= 8	false
<=	less than or equal to	x <= 8	true

Dynamic types

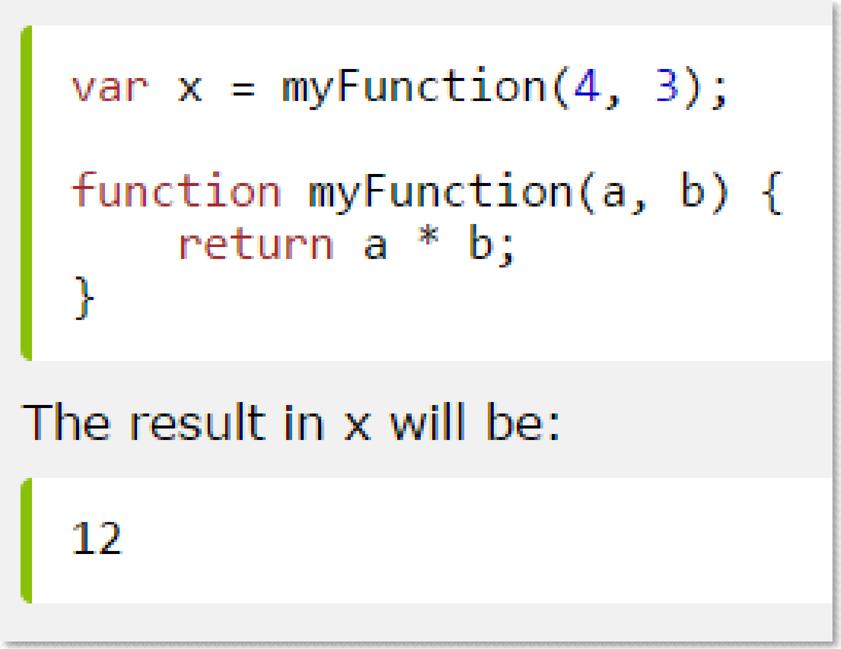


```
typeof "John"                // Returns string
typeof 3.14                  // Returns number
typeof false                 // Returns boolean
typeof [1,2,3,4]             // Returns object
typeof {name:'John', age:34} // Returns object
```



```
var person; // The value is undefined, the typeof is undefined
```

Functions



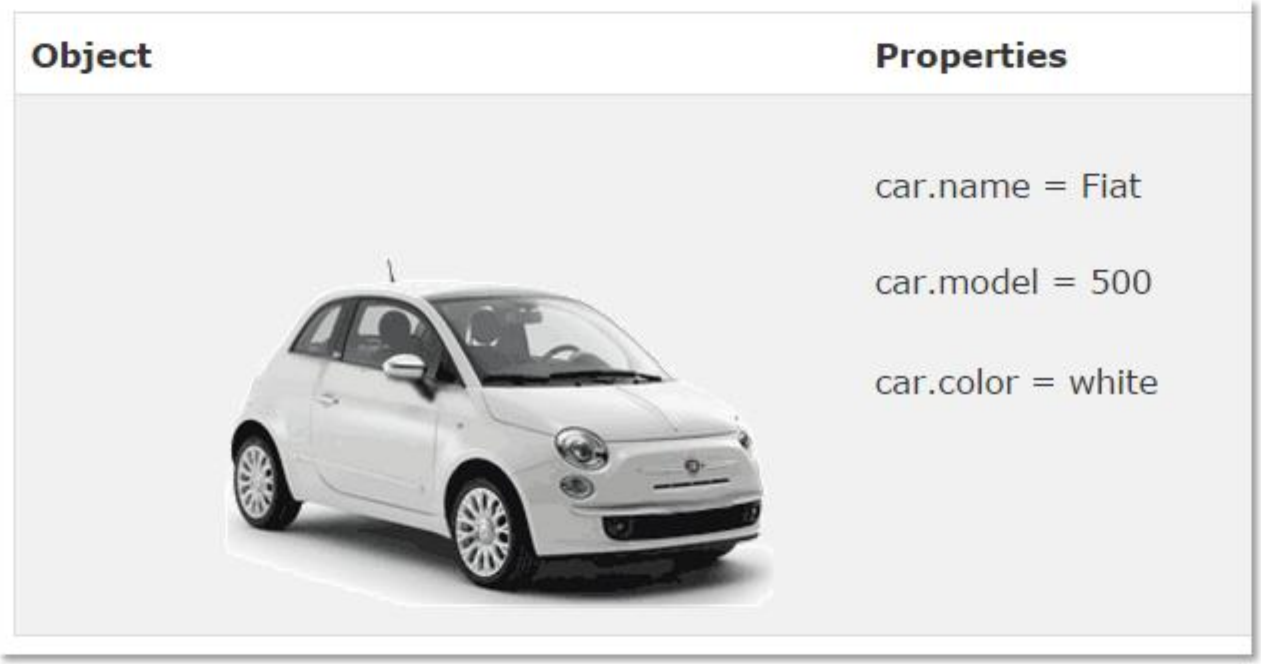
```
var x = myFunction(4, 3);

function myFunction(a, b) {
    return a * b;
}
```

The result in x will be:

```
12
```

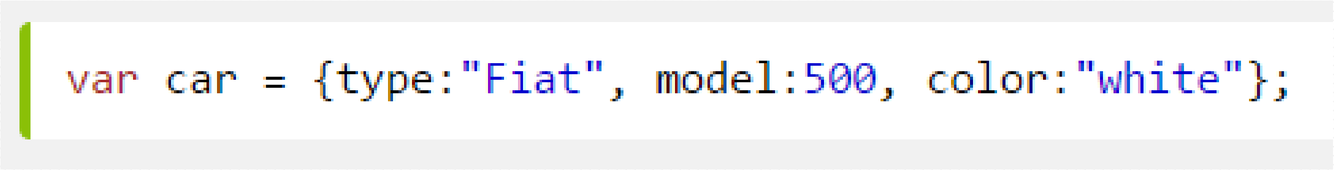
Objects

Objects are variables too.
But **objects** can contain many values.



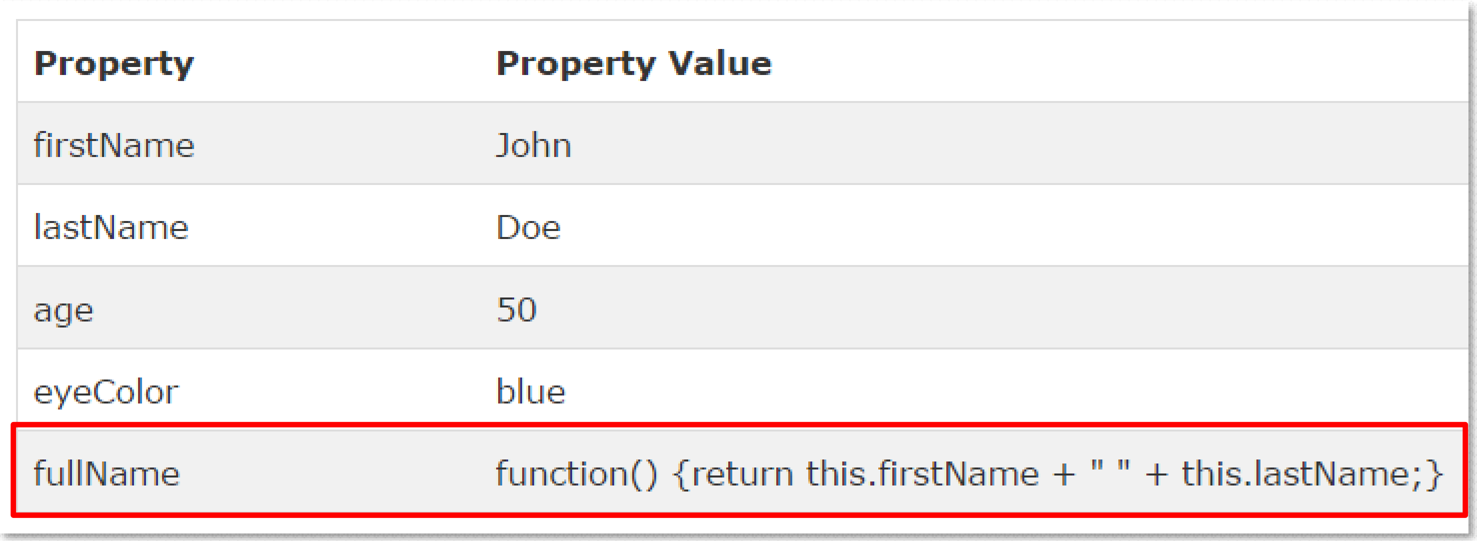
Object	Properties

	car.name = Fiat
	car.model = 500
	car.color = white

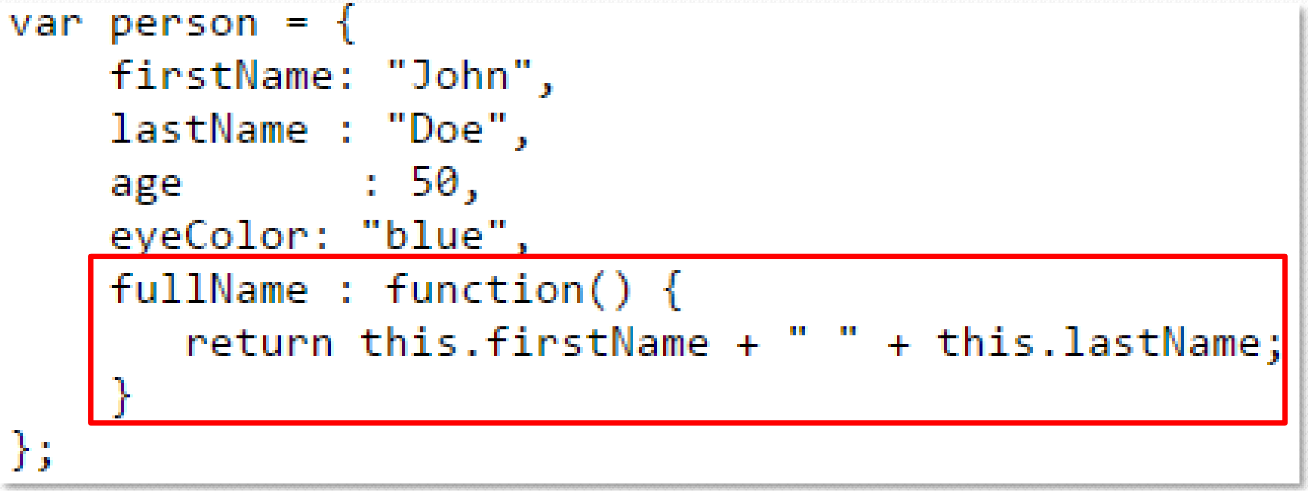


```
var car = {type:"Fiat", model:500, color:"white"};
```

Objects

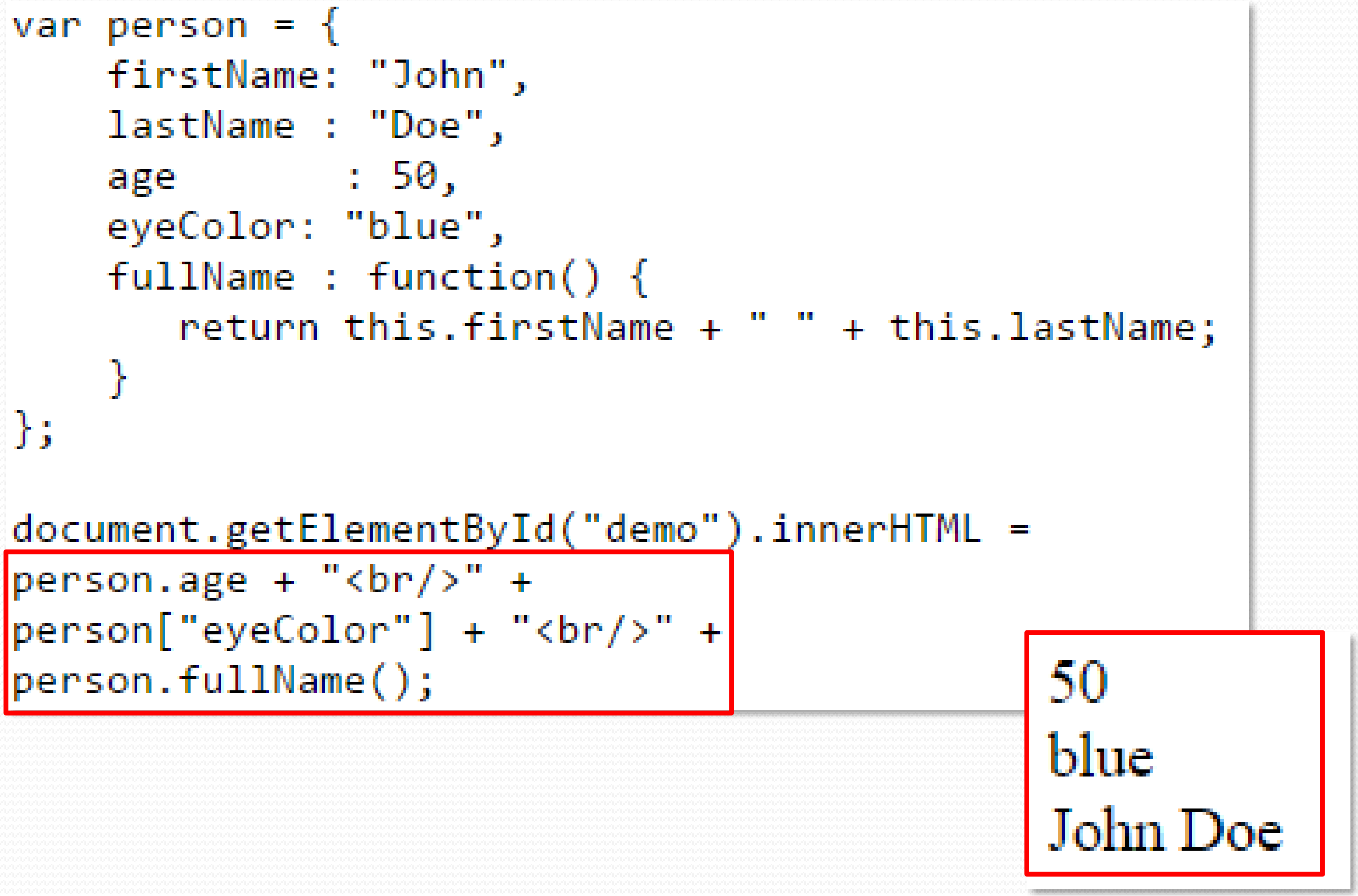


Property	Property Value
firstName	John
lastName	Doe
age	50
eyeColor	blue
fullName	function() {return this.firstName + " " + this.lastName;}



```
var person = {
    firstName: "John",
    lastName : "Doe",
    age      : 50,
    eyeColor: "blue",
    fullName : function() {
       return this.firstName + " " + this.lastName;
    }
};
```

Objects



```
var person = {
    firstName: "John",
    lastName : "Doe",
    age       : 50,
    eyeColor: "blue",
    fullName : function() {
       return this.firstName + " " + this.lastName;
    }
};

document.getElementById("demo").innerHTML =
person.age + "<br/>" +
person["eyeColor"] + "<br/>" +
person.fullName();
```

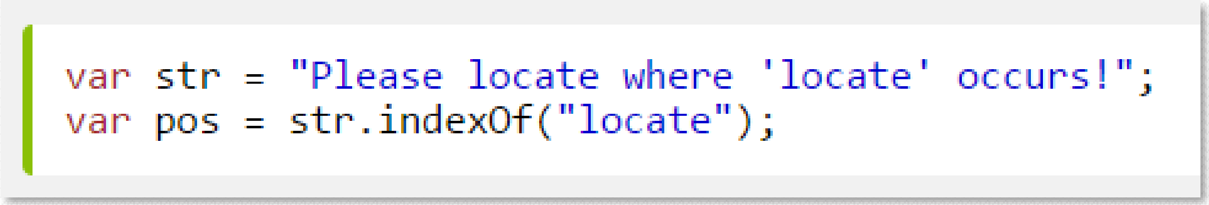
```
50
blue
John Doe
```

Common events

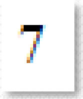
Nazwa	Opis
onchange	An HTML element has been changed
onclick	The user clicks an HTML element
onmouseover	The user moves the mouse over an HTML element
onmouseout	The user moves the mouse away from an HTML element
onkeydown	The user pushes a keyboard key
onload	The browser has finished loading the page

String type methods

indexOf()

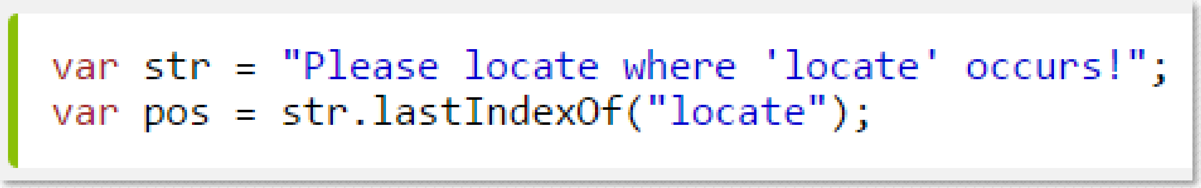


```
var str = "Please locate where 'locate' occurs!";
var pos = str.indexOf("locate");
```

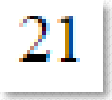


```
7
```

lastIndexOf()

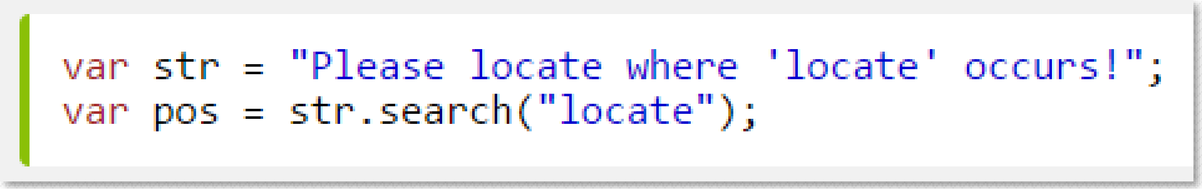


```
var str = "Please locate where 'locate' occurs!";
var pos = str.lastIndexOf("locate");
```

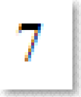


```
21
```

search()



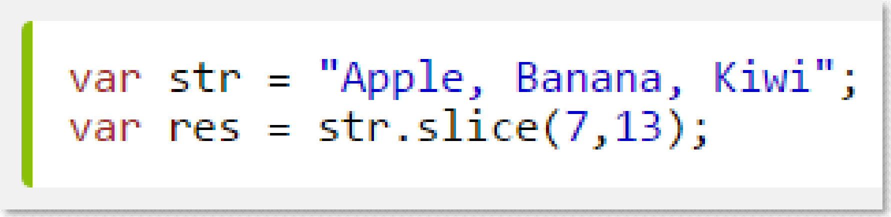
```
var str = "Please locate where 'locate' occurs!";
var pos = str.search("locate");
```



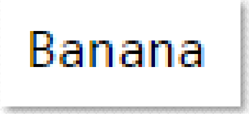
```
7
```

String type methods

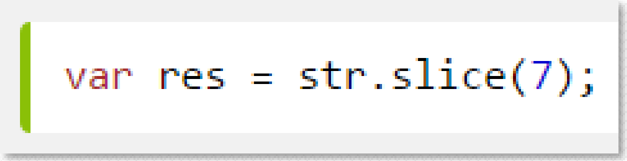
slice()



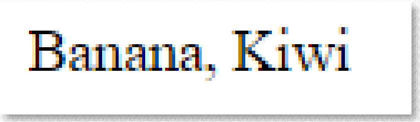
```
var str = "Apple, Banana, Kiwi";
var res = str.slice(7,13);
```



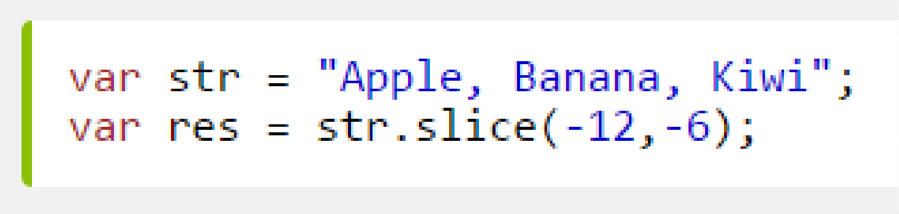
```
Banana
```



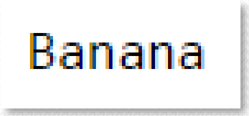
```
var res = str.slice(7);
```



```
Banana, Kiwi
```



```
var str = "Apple, Banana, Kiwi";
var res = str.slice(-12,-6);
```

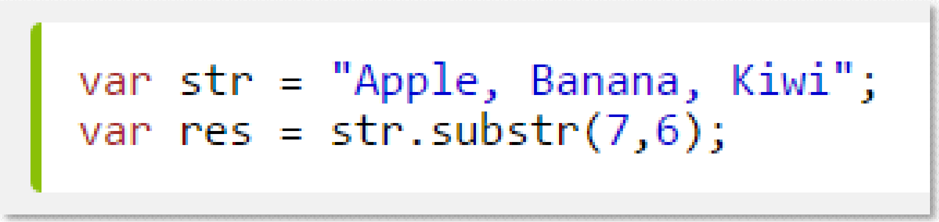


```
Banana
```

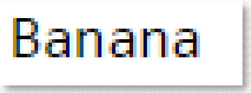
substring() – works like slice(),
but **it does not accept negative indexes**

String type methods

substr()

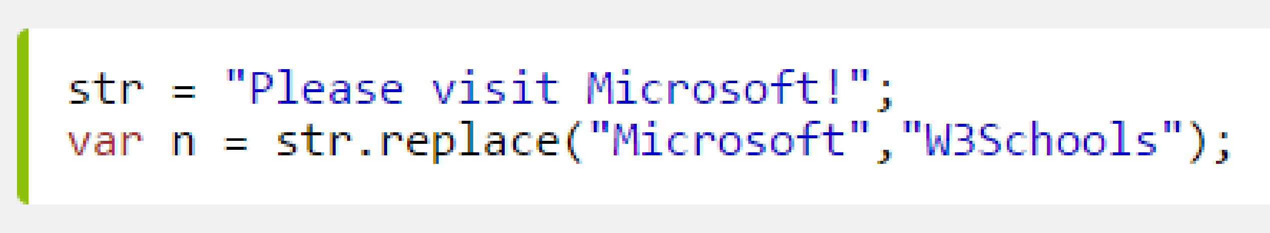


```
var str = "Apple, Banana, Kiwi";
var res = str.substr(7,6);
```

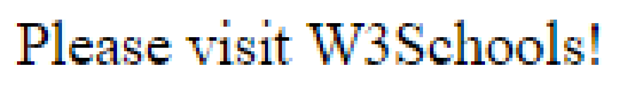


```
Banana
```

replace()



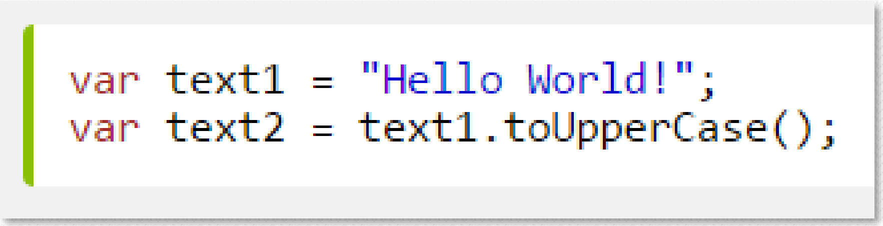
```
str = "Please visit Microsoft!";
var n = str.replace("Microsoft","W3Schools");
```



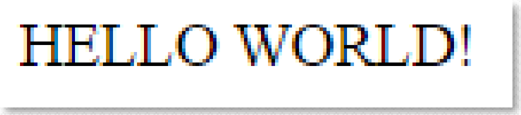
```
Please visit W3Schools!
```

String type methods

toUpperCase()

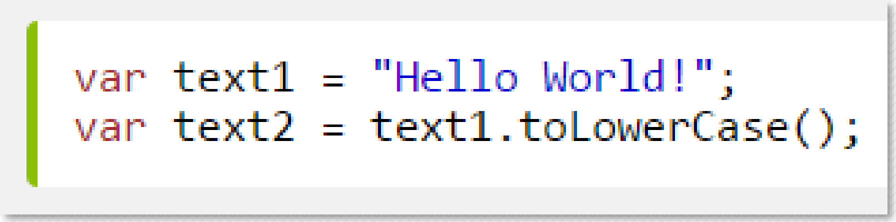


```
var text1 = "Hello World!";
var text2 = text1.toUpperCase();
```

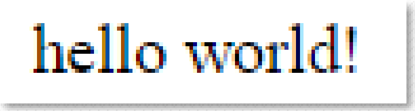


HELLO WORLD!

toLowerCase()



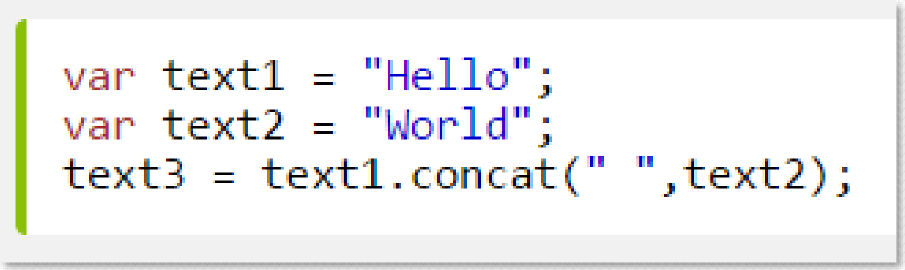
```
var text1 = "Hello World!";
var text2 = text1.toLowerCase();
```



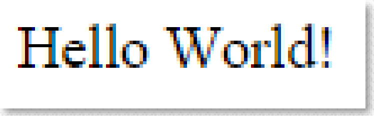
hello world!

String type methods

concat()

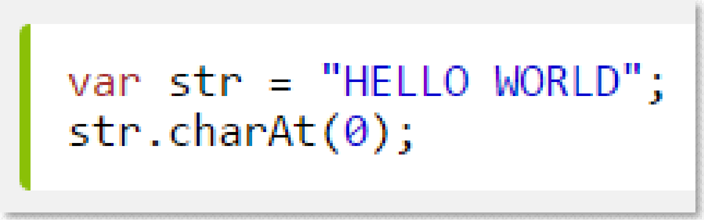


```
var text1 = "Hello";
var text2 = "World";
text3 = text1.concat(" ",text2);
```

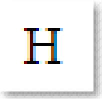


```
Hello World!
```

charAt()

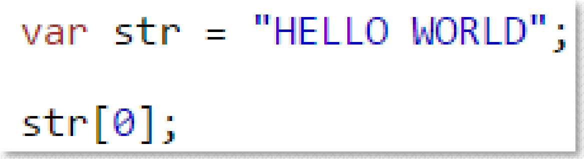


```
var str = "HELLO WORLD";
str.charAt(0);
```



```
H
```

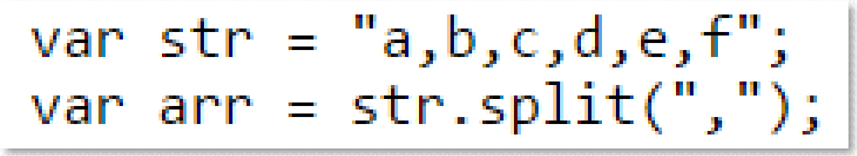
Using the following instruction **is not safe!**



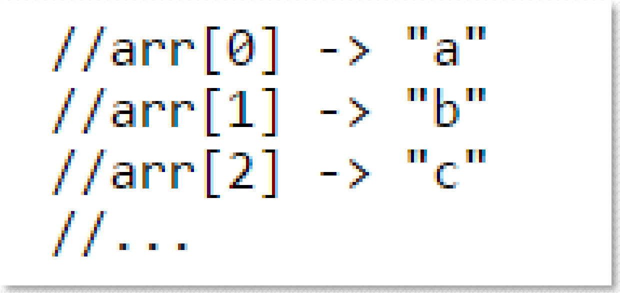
```
var str = "HELLO WORLD";

str[0];
```

String type methods

split()



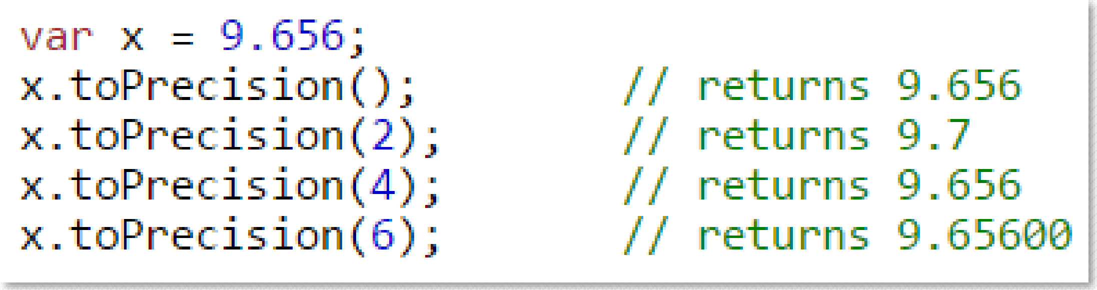
```
var str = "a,b,c,d,e,f";
var arr = str.split(",");
```



```
//arr[0] -> "a"
//arr[1] -> "b"
//arr[2] -> "c"
//...
```

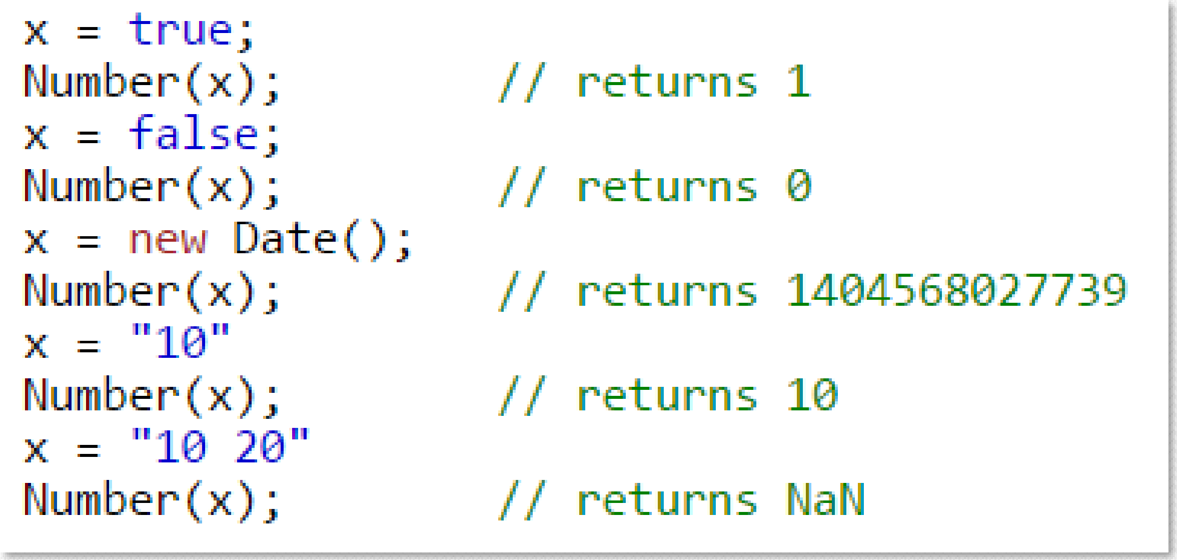
Numeric type methods

toPrecision()



```
var x = 9.656;
x.toPrecision();          // returns 9.656
x.toPrecision(2);         // returns 9.7
x.toPrecision(4);         // returns 9.656
x.toPrecision(6);         // returns 9.65600
```

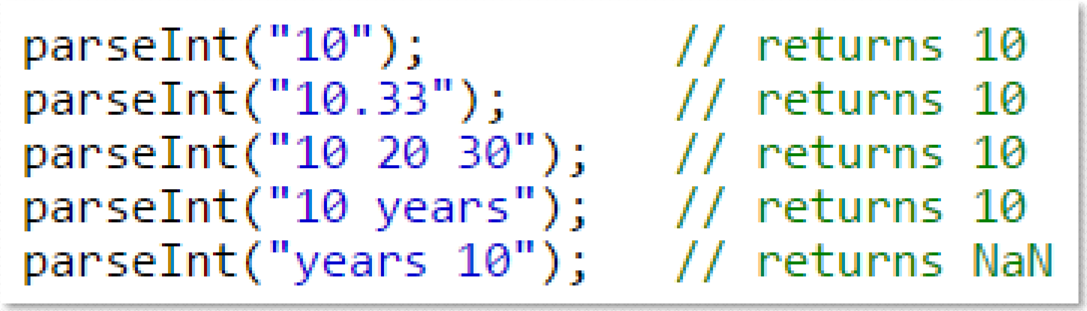
Number()



```
x = true;
Number(x);        // returns 1
x = false;
Number(x);        // returns 0
x = new Date();
Number(x);        // returns 1404568027739
x = "10"
Number(x);        // returns 10
x = "10 20"
Number(x);        // returns NaN
```

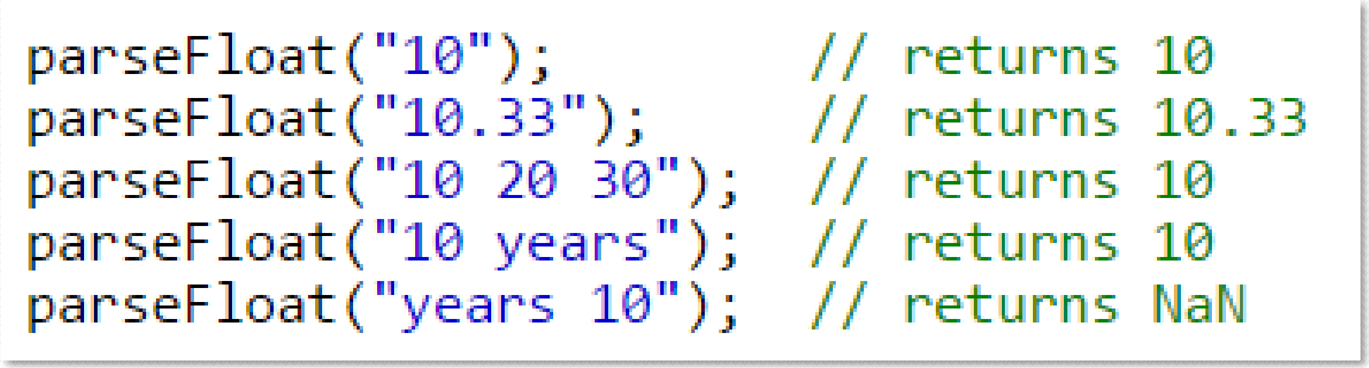
Numeric type methods

parseInt()



```
parseInt("10");          // returns 10
parseInt("10.33");       // returns 10
parseInt("10 20 30");    // returns 10
parseInt("10 years");    // returns 10
parseInt("years 10");    // returns NaN
```

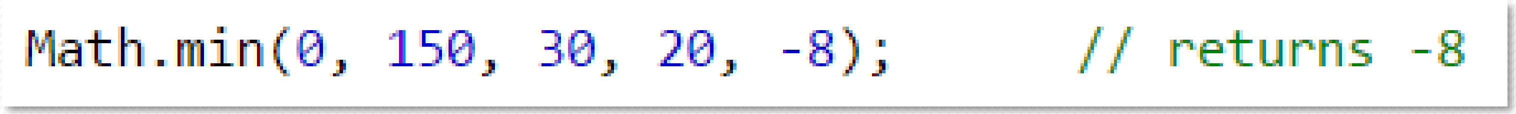
parseFloat()



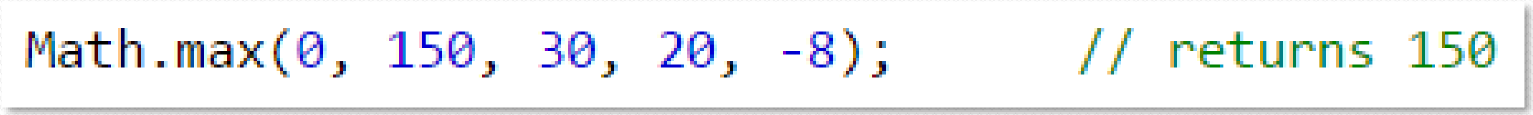
```
parseFloat("10");          // returns 10
parseFloat("10.33");       // returns 10.33
parseFloat("10 20 30");    // returns 10
parseFloat("10 years");    // returns 10
parseFloat("years 10");    // returns NaN
```

Mathematic methods

Math.min() i Math.max()

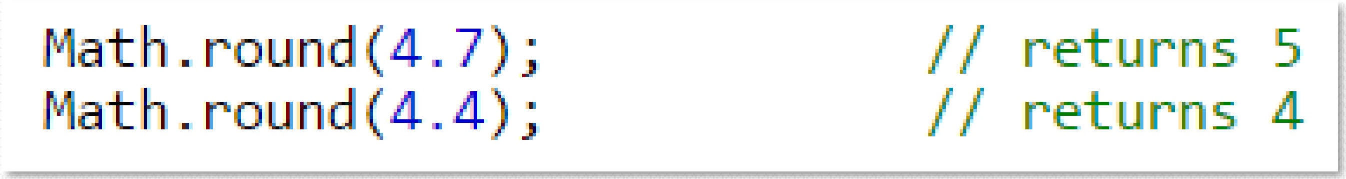


```
Math.min(0, 150, 30, 20, -8);       // returns -8
```



```
Math.max(0, 150, 30, 20, -8);       // returns 150
```

Math.round()

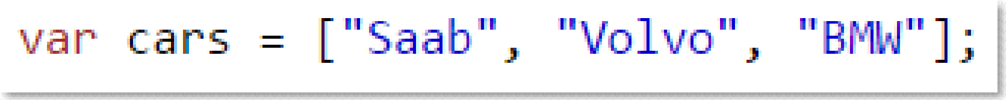


```
Math.round(4.7);              // returns 5
Math.round(4.4);              // returns 4
```

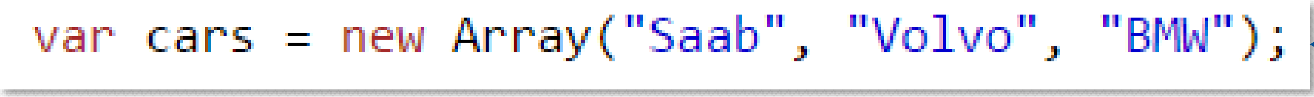
And other, e.g. random(), sin(), abs(), sqrt(), log(), …

Arrays

Creation



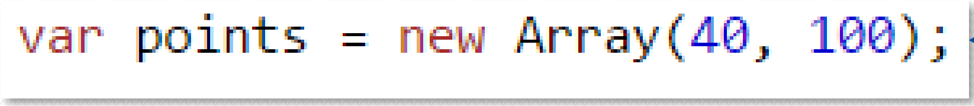
```
var cars = ["Saab", "Volvo", "BMW"];
```



```
var cars = new Array("Saab", "Volvo", "BMW");
```

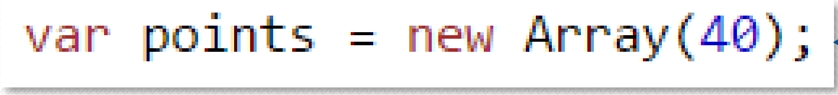
← Not recommended

Example



```
var points = new Array(40, 100);
```

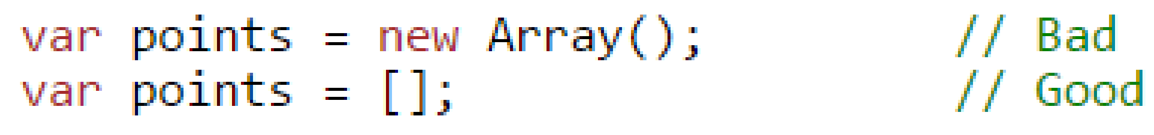
← 2 elementw – „40”, „100”



```
var points = new Array(40);
```

← 40 empty elements

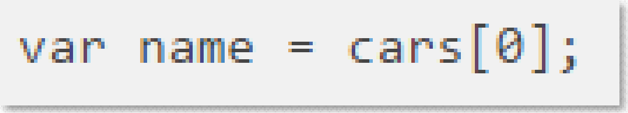
Creation of empty array



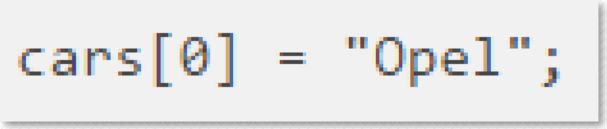
```
var points = new Array();             // Bad
var points = [];                      // Good
```

Arrays

Access to the elements

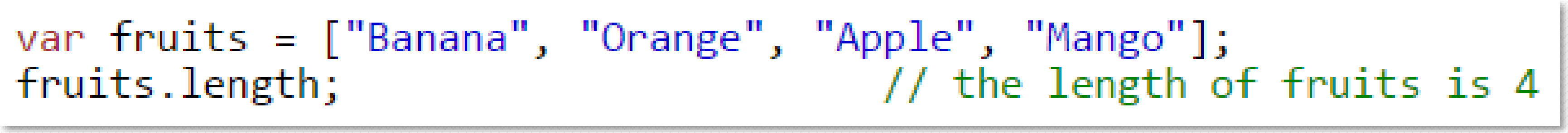


```
var name = cars[0];
```



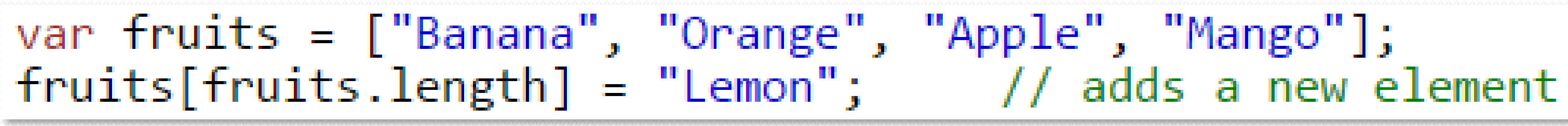
```
cars[0] = "Opel";
```

Number of elements



```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.length;                      // the length of fruits is 4
```

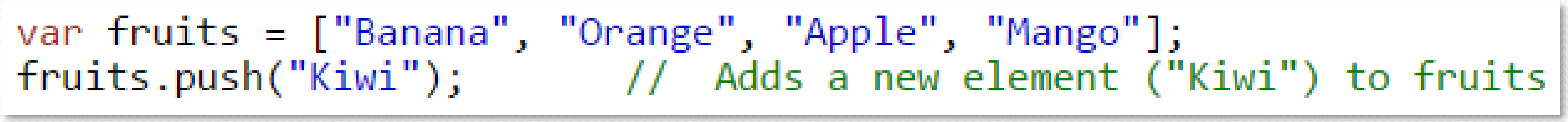
Adding of new element



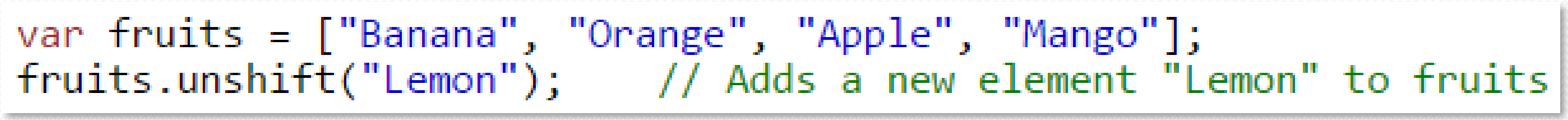
```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits[fruits.length] = "Lemon";     // adds a new element
```

Arrays

Adding of the element

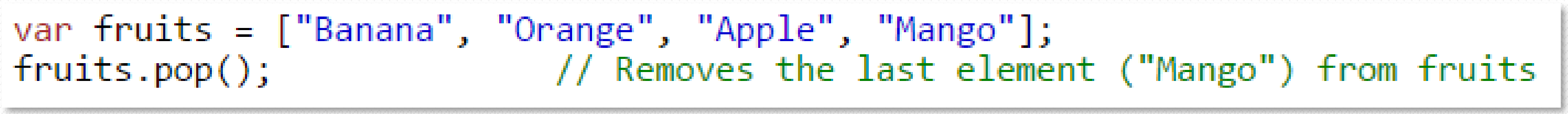


```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.push("Kiwi");          //  Adds a new element ("Kiwi") to fruits
```

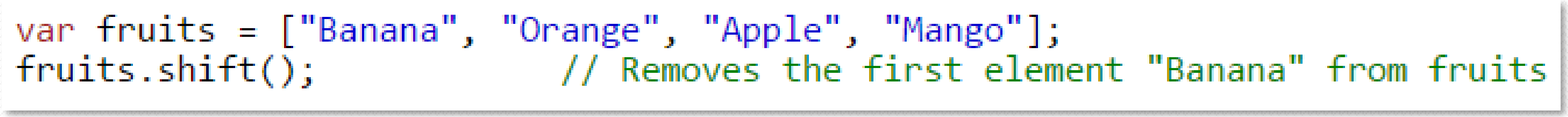


```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.unshift("Lemon");     // Adds a new element "Lemon" to fruits
```

Removing of the element



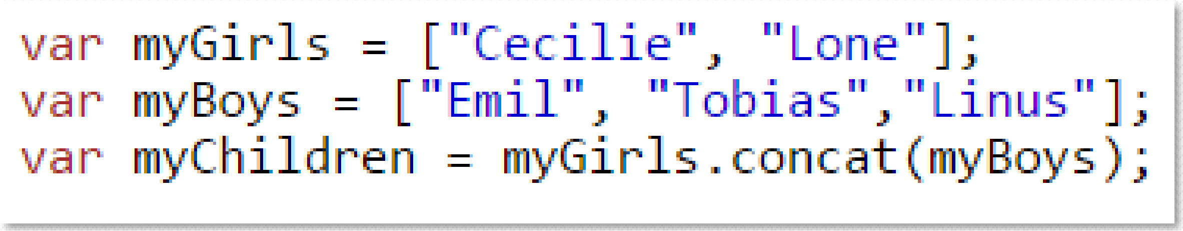
```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.pop();                // Removes the last element ("Mango") from fruits
```



```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.shift();              // Removes the first element "Banana" from fruits
```

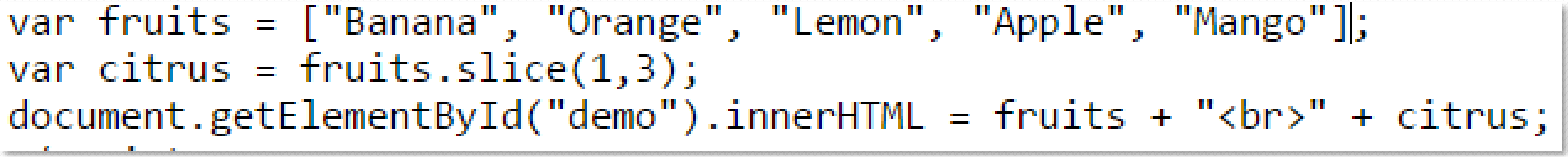
Arrays

Arrays concatenation

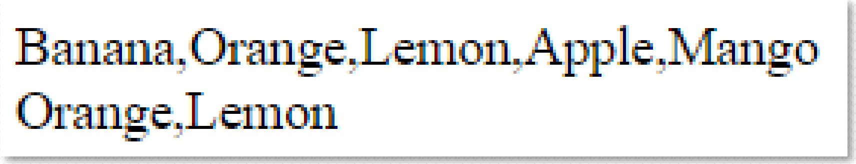


```
var myGirls = ["Cecilie", "Lone"];
var myBoys = ["Emil", "Tobias","Linus"];
var myChildren = myGirls.concat(myBoys);
```

Slicing of an array fragment

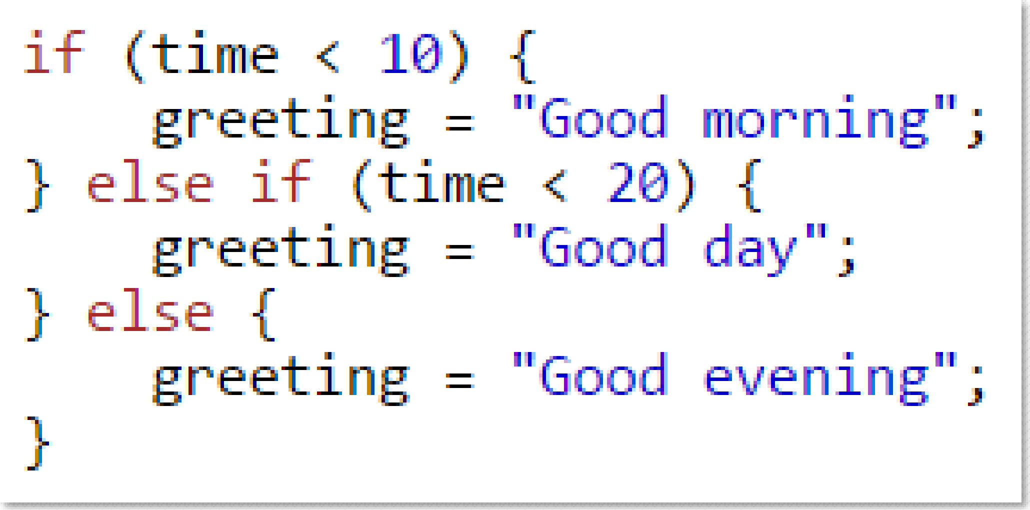


```
var fruits = ["Banana", "Orange", "Lemon", "Apple", "Mango"];
var citrus = fruits.slice(1,3);
document.getElementById("demo").innerHTML = fruits + "<br>" + citrus;
```



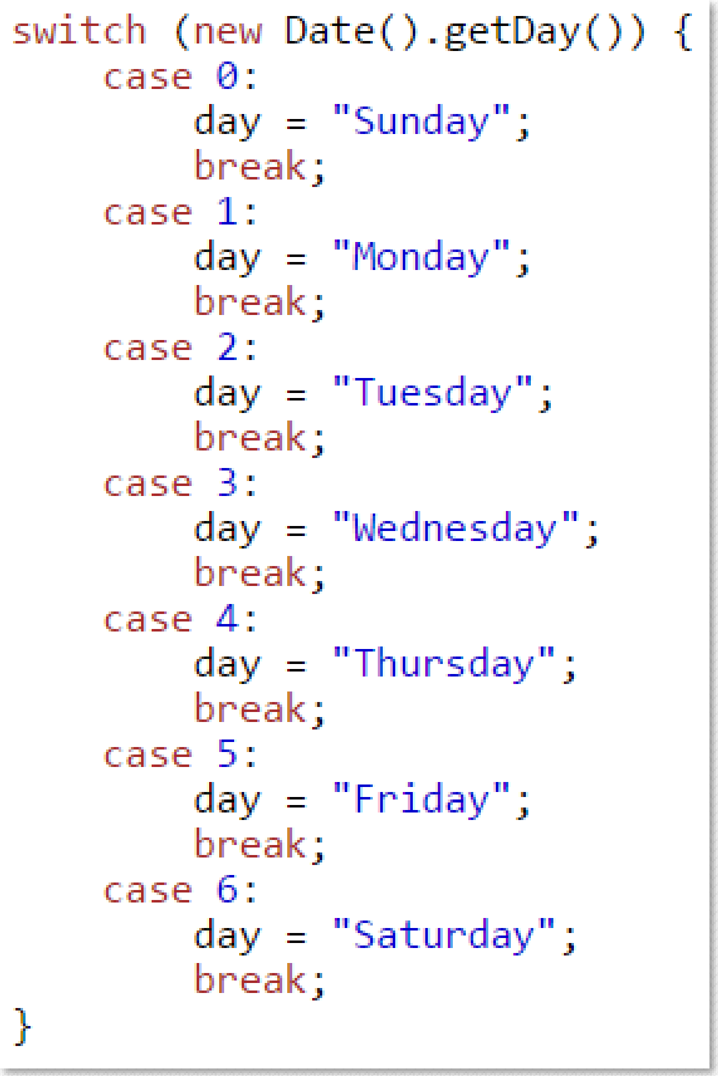
```
Banana,Orange,Lemon,Apple,Mango
Orange,Lemon
```

Conditional instructions



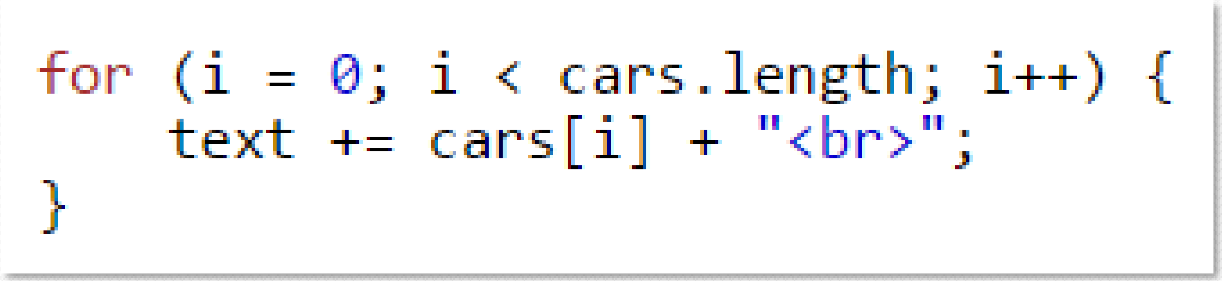
```
if (time < 10) {
    greeting = "Good morning";
} else if (time < 20) {
    greeting = "Good day";
} else {
    greeting = "Good evening";
}
```

Conditional instructions



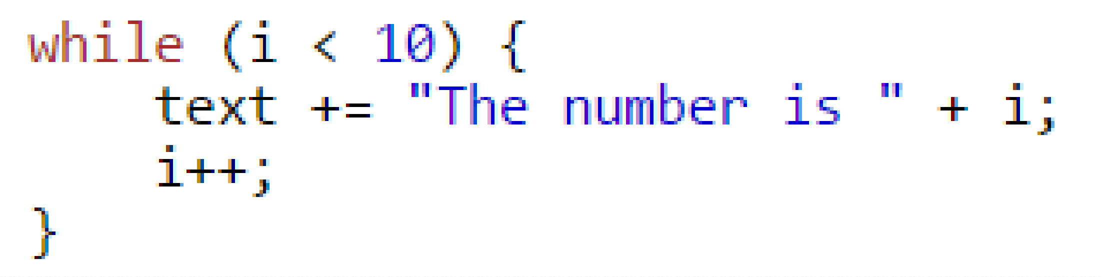
```
switch (new Date().getDay()) {
    case 0:
        day = "Sunday";
        break;
    case 1:
        day = "Monday";
        break;
    case 2:
        day = "Tuesday";
        break;
    case 3:
        day = "Wednesday";
        break;
    case 4:
        day = "Thursday";
        break;
    case 5:
        day = "Friday";
        break;
    case 6:
        day = "Saturday";
        break;
}
```

For loop



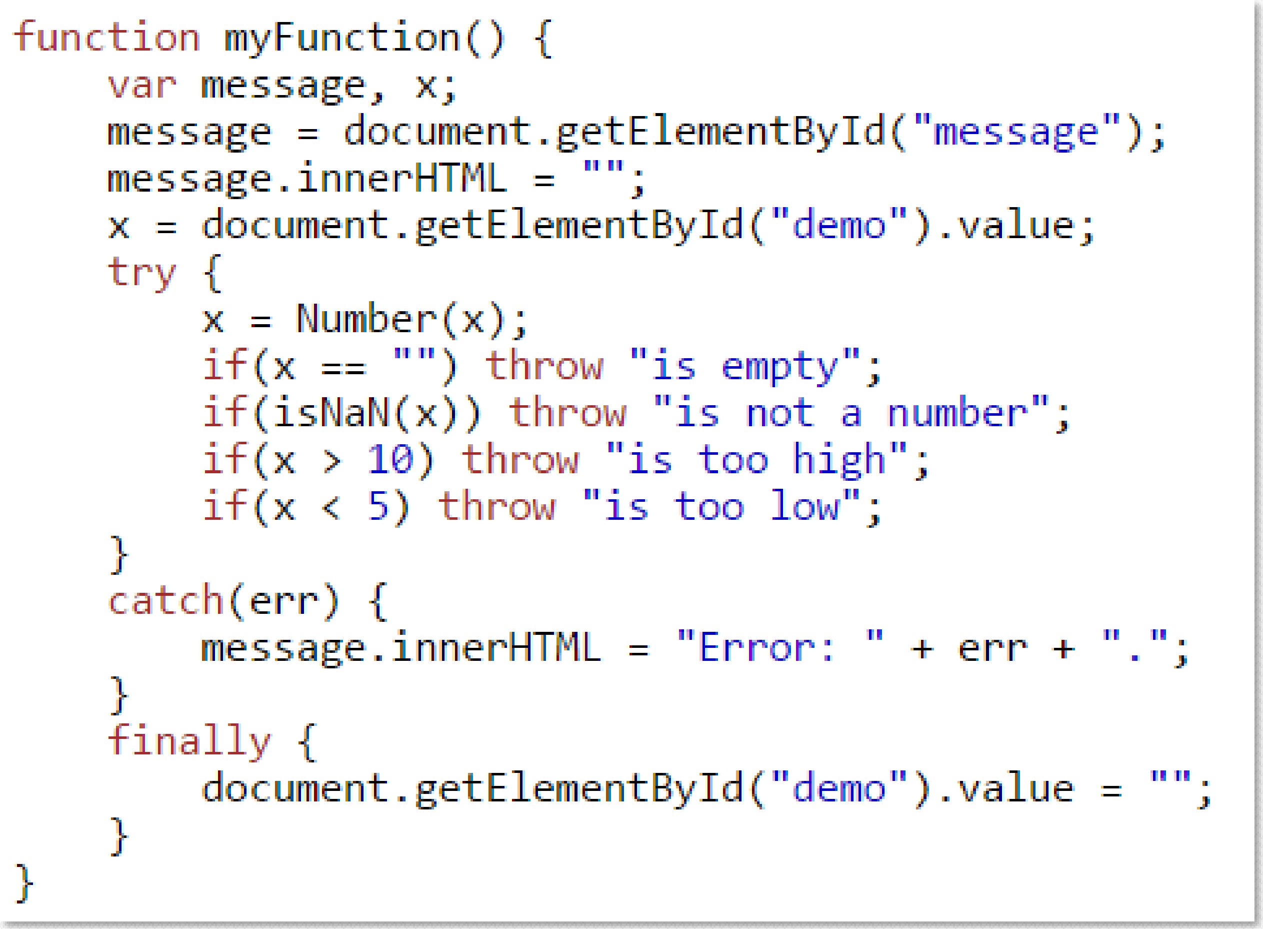
```
for (i = 0; i < cars.length; i++) {
    text += cars[i] + "<br>";
}
```

While loop



```
while (i < 10) {
    text += "The number is " + i;
    i++;
}
```

Exception handling



```
function myFunction() {
    var message, x;
    message = document.getElementById("message");
    message.innerHTML = "";
    x = document.getElementById("demo").value;
    try {
        x = Number(x);
        if(x == "") throw "is empty";
        if(isNaN(x)) throw "is not a number";
        if(x > 10) throw "is too high";
        if(x < 5) throw "is too low";
    }
    catch(err) {
        message.innerHTML = "Error: " + err + ".";
    }
    finally {
        document.getElementById("demo").value = "";
    }
}
```