

JavaScript

intro

PHP web development 2019/2020

Milena Tomova
Vratsa Software

<https://vratsasoftware.com/>

Table of Contents

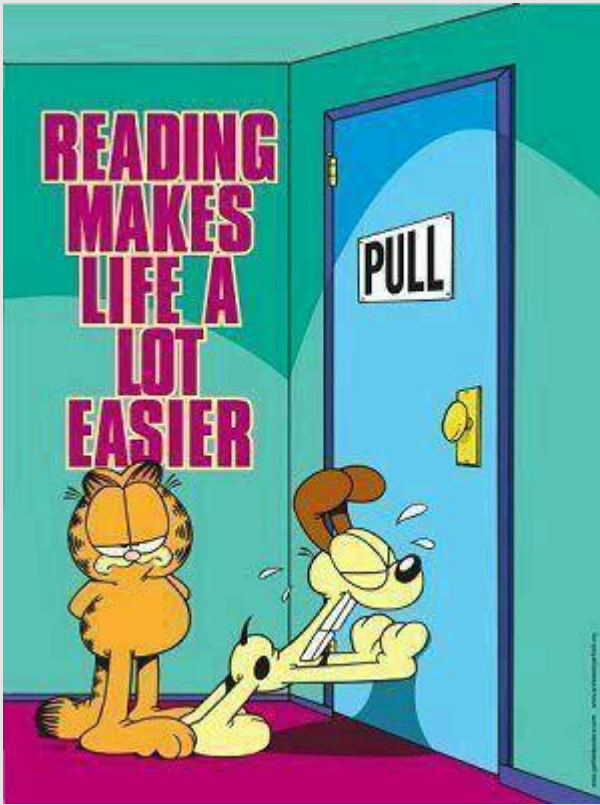


1. Resources
2. Projects
3. Development Tools
4. What can/not/ JavaScript Do?
5. JS up to now
6. How to add JS to your projects?
7. JS syntax
8. Data types and variables
9. JS specials - NaN, undefined, Null
10. Hoisting



resources

resources



MDN

Murach's JavaScript, Nth Edition, Training & Reference

Murach's JavaScript and jQuery, Nth Edition, Training & Reference

Beyond jQuery, Ray Nicholus



Projects

projects

Projects


1. One man task
2. The stress is on logic
3. Solved using only JavaScript and jQuery
4. The solution should work with different input data
5. Using GitHub during development will be a must.

Don't try to understand
~~JavaScript~~ girl completely. If you do,
then either you'll go mad..
or you'll start loving ~~it~~ her.

- Unknown



Development Tools



What can/not/
JavaScript Do?

What can/not/ JavaScript Do?

JavaScript

- JavaScript is a scripting language that is run by the JavaScript engine of a web browser and controls the operation of the browser.
- Can control DOM events
- Can traverse and create HTML elements
- Can modify DOM tree
- Can access and/or modify browser's cookies
- Can perform **asynchronous** request, best known as AJAX

What can/not/ JavaScript Do?

JavaScript

turn off JavaScript on your browser and see what happens



JavaScript in time



JavaScript in time

JavaScript

Ver	Official Name	Description
1	ECMAScript 1 (1997)	First Edition.
2	ECMAScript 2 (1998)	Editorial changes only.
3	ECMAScript 3 (1999)	Added Regular Expressions. Added try/catch.
4	ECMAScript 4	Never released.
5	ECMAScript 5 (2009) Read More: JS ES5	Added "strict mode". Added JSON support. Added String.trim(). Added Array.isArray(). Added Array Iteration Methods.
5.1	ECMAScript 5.1 (2011)	Editorial changes.
6	ECMAScript 2015 Read More: JS ES6	Added let and const. Added default parameter values. Added Array.find(). Added Array.findIndex().
7	ECMAScript 2016	Added exponential operator (**). Added Array.prototype.includes.
8	ECMAScript 2017	Added string padding. Added new Object properties. Added Async functions. Added Shared Memory.
9	ECMAScript 2018	Added rest / spread properties. Added Asynchronous iteration. Added Promise.finally(). Additions to RegExp.



How to add JS to your
projects?

How to add JS to

```
<body>  
  <script type="text/javascript">  
    alert('Hello JavaScript!');  
  </script>  
</body>
```

Saving uploaded file

Script in the head element

```
<script type="text/javascript">  
    //js script goes here;  
</script>
```

External files linked in the head element

```
<script src="path/to/your/js/file/here" type="text/javascript"></script>
```


Saving uploaded file

Script in the body element

```
<script type="text/javascript">  
    //js script goes here;  
</script>
```

External files linked before the closing body tag

```
<script src="path/to/your/js/file/here" type="text/javascript"></script>  
</body>
```




JS syntax


Remember

1. Variables and function names are Case-sensitive
2. Every declaration should end with ;
3. Whitespace/tab is ignored
4. JS code can be carried over to a new line for better readability under certain conditions. / JS will try to add ; if he considers that they are missing 😊 /



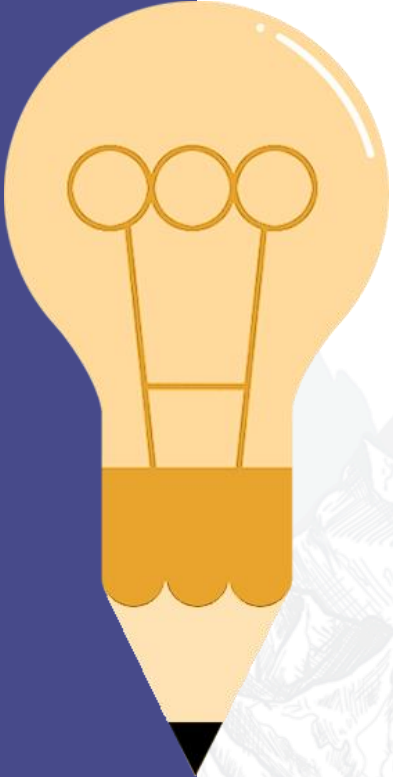
Remember

Basic rules for naming variables

- 
1. Can contain letters, numbers, _, \$ / don't confuse it with \$ in PHP /
 2. Cannot start with a number
 3. Variable names are case-sensitive.
 4. There is no limit to the number of characters
but still DO NOT name them this way -
theManOfTheGameThatPlayedTheDayBeforeYesterday! :))
 1. There are special words in JS that cannot be used as variable
names - new, this, etc.

Remember

Basic rules for naming variables

- 
6. More popular for JS is the camel casing (taxRate)
if you use underscore for word separation -
be consistent in your code
do not mixed both

Remember

Basic rules - comments

`/* Block comments */`

`//Single-line comments`

`//Single-line comments`

`//Single-line comments`



Remember

Everything in JavaScript is a predefined object

Every string is a JS object

Every number is a JS object

Every array is a JS object

Every object is a JS object

Every JS object comes with a predefined behaviour - a set of methods/functions and properties that can be used to write JS programs.



Remember

Everything in JavaScript is a predefined object

Often will get an error that says *... is not a function ...*

If you've written the function correctly /no syntax mistake/ that will definitely mean that you are trying to use a function that do not belong to the type of object you have - a function that belongs to a string object and not to a number object for example!



Window is a global JavaScript object, that refers to browser window
`alert` is window's method

```
window.alert( "This is a test of the alert method " ) ;
```

Notice You can omit window, because it is a global object, when calling its alert method

```
alert( "This is a test of the alert method " ) ;
```


Remember

How to call JS object method

`window.alert()`

How to use JS object property

`alert(window.location)`

*will print the URL of the current open page
location is a window property*





JS variables

JS variables

```
var subtotal; // declares one variable  
var investment, interestRate, years; // declares three variables
```

Declaring a variable and setting a value

```
var subtotal = 74.00, salesTax;  
salesTax = subtotal *.1; //subtotal = 74.00, salesTax = 7.4
```

After declaring the variable, its subsequent use is without var!

JS variables

Set a value with an expression

```
var subtotal = 74.95; //subtotal = 74.95  
subtotal += 20.00; // subtotal = 94.95
```




JS datatypes

Remember

Datatypes in JS



Like in PHP, we do not explicitly declare the type of data that the variable will store.

The value we give to the variable can be of type

Integer

Fractional number - float

String

Array

Boolean

NULL

Remember

Datatypes in JS

Specific for JS are undefined and NaN



Remember

Operators in JS

The only difference from PHP is the behaviour
of + operator



JS syntax - operators

```
console.log( 1 + 1 ) //results is 2
```

```
console.log( '1' + '1' ) //results is 11
```

```
console.log( '1' + 1 ) //results is 11
```


JS syntax - operators

Be careful when using JavaScript for calculations with a floating-point numbers

```
var subtotal = 74.95, salesTax; // subtotal = 74.95  
    salesTax = subtotal * . 1; //salesTax = 7.4950000000000001
```


Remember

`parseInt ()` & `parseFloat ()`



`parseInt (string)`

Converts a string to an integer and returns the resulting value.
If the string is not convertible to a number, returns **NaN**.

`parseFloat (string)`

Converts a string to a float and returns the resulting value.
If the string is not convertible to a float, returns NaN.

NaN = not a number

JS syntax - operators

Fastest way to convert a string to a number/to cast to a number is to use +

```
var subtotal = '74.95'; // string  
subtotal = +subtotal + 10; //84.95
```


Remember

`typeof(var)` check the variable type



JS syntax - typeof(var)

```
var x = 5;  
console.log(typeof(x)); // number  
console.log(x); // 5  
  
x = new Number(5);  
console.log(typeof(x)); // object  
console.log(x); // Number {}  
  
x = null;  
console.log(typeof(x)); // object  
  
x = undefined;  
console.log(typeof(x)); // undefined
```



hoisting

hoisting

hoisting

Hoisting is JavaScript's default behavior of moving declarations to the top.

JavaScript Declarations are Hoisted

In JavaScript, a variable can be declared after it has been used.

In other words; a variable can be used before it has been declared.

JS syntax - hoisting

```
x = 5; // Assign 5 to x  
console.log(x) //5  
var x; // Declare x
```

hoisting

hoisting

JavaScript Initializations are Not Hoisted

JavaScript only hoists declarations, not initializations.

JS syntax - hoisting

```
var x = 5; // Initialize x  
var y = 7; // Initialize y  
console.log(x); //5  
console.log(y); //7
```

```
var x = 5; // Initialize x  
console.log(x); //5  
console.log(y); //undefined  
var y = 7; // Initialize y
```


Remember

Declare Your Variables At the Top !

Hoisting is (to many developers) an unknown or overlooked behavior of JavaScript.

If a developer doesn't understand hoisting, programs may contain bugs (errors).

To avoid bugs, always declare all variables at the beginning of every scope.

Since this is how JavaScript interprets the code, it is always a good rule.



JS syntax - hoisting

```
var x = 5, y = 7; // Initialize x, y  
console.log(x); //5  
console.log(y); //7
```

```
var x, y; // declare x, y  
  
x = 5; // assign value to x  
y = 7; // assign value to x  
  
console.log(x); //5  
console.log(y); //7
```


Questions?



Гнездото
Coworking

Цялостен
курс по
програми
ране

Дизайн
курс

Курс по
дигит.
маркетинг

MindHub



Partners



**Telerik
Academy**



Trainings @ Vratsa Software



- Vratsa Software – High-Quality Education, Profession and Jobs
 - www.vratsasoftware.com
- The Nest Coworking
 - www.nest.bg
- Vratsa Software @ Facebook
 - www.fb.com/VratsaSoftware
- Slack Channel
 - www.vso.slack.com

