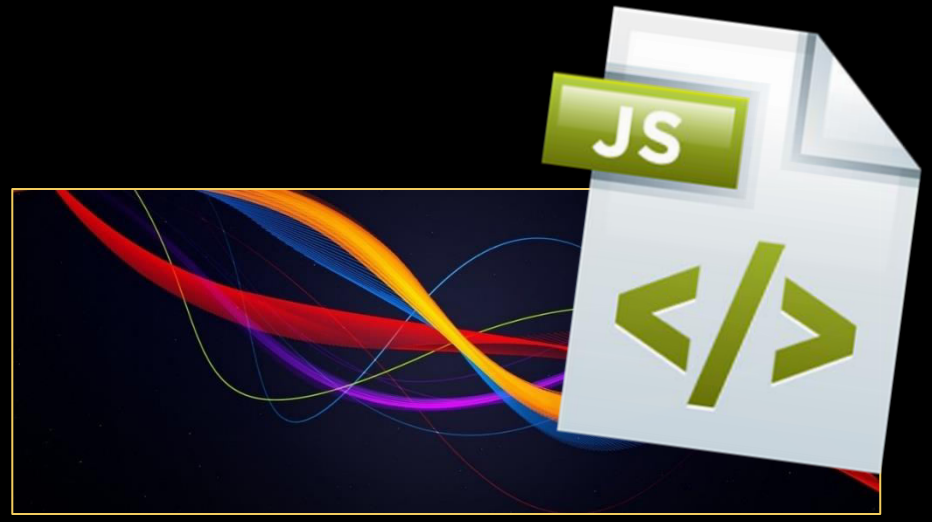
JavaScript Syntax

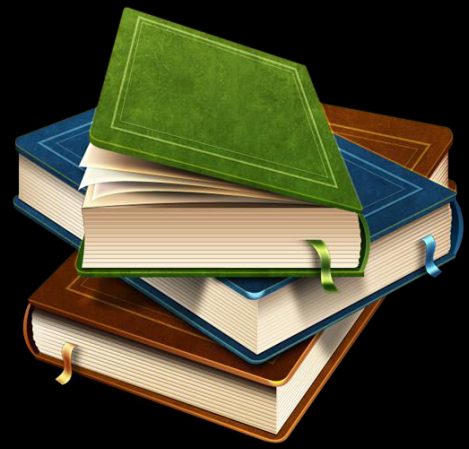
Syntax, Conditions, Loops, Functions, Objects, Arrays



SoftUni Team
Technical Trainers

Software University
http://softuni.bg

Table of Contents



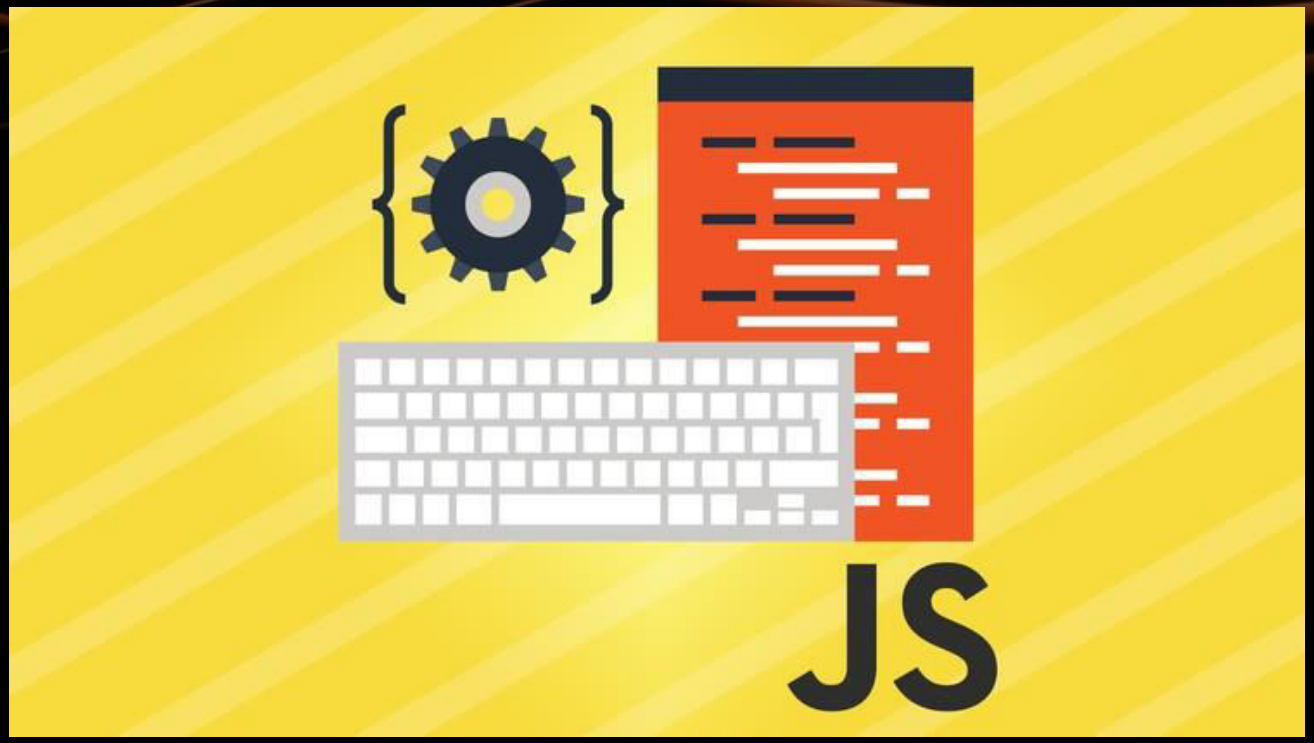
Have a Question?

SoftUni Foundation

sli.do

#front-end

SoftUni Foundation



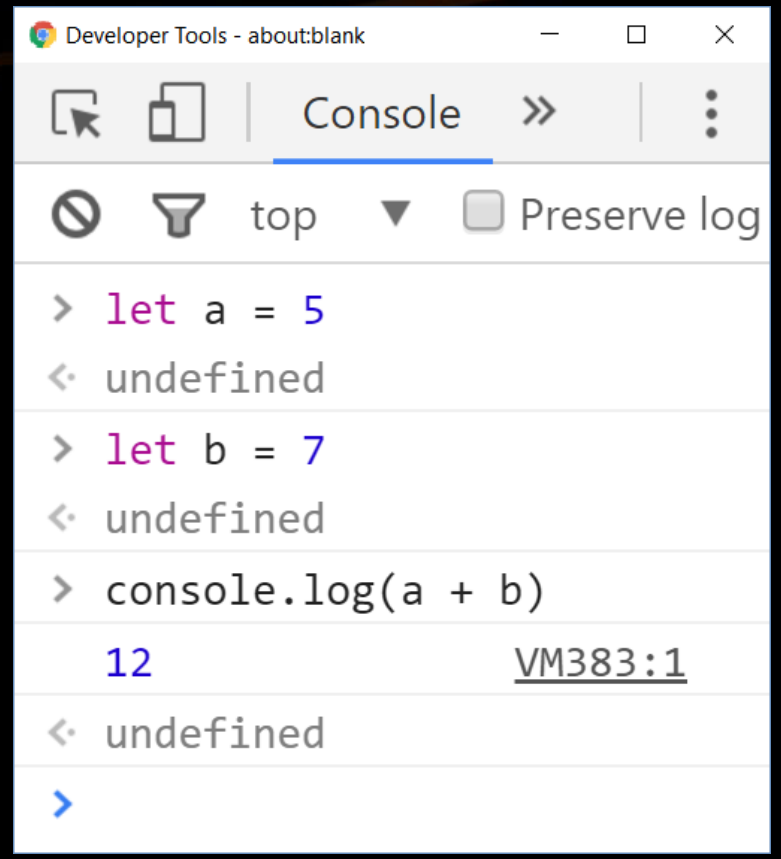
JavaScript Introduction

Syntax and Comparison

Welcome to JavaScript

SoftUni Foundation

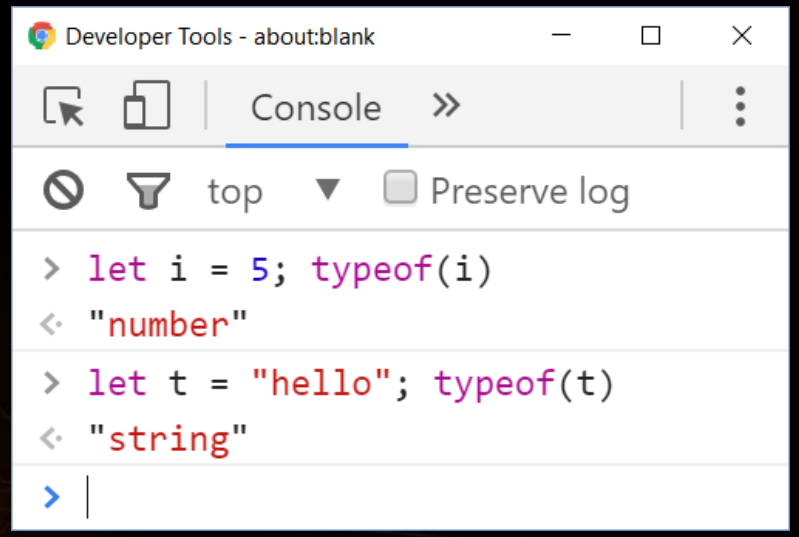
- JavaScript (JS) is a **scripting language**
  - Executes commands (script)
  - Can work in interactive mode
  - No compilation, just execute commands
- C# / Java are **statically-typed** languages
  - Programs are first compiled
  - Then the compiled binary is executed

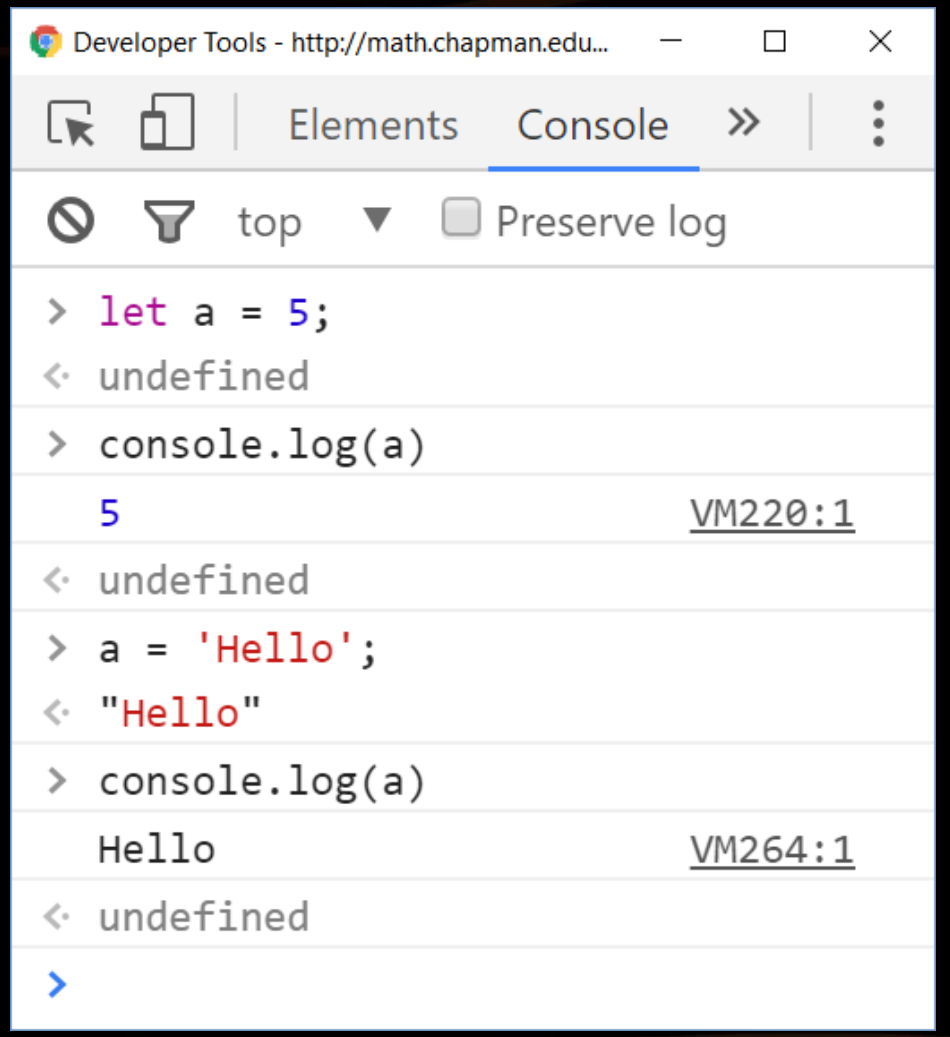


Welcome to JavaScript (2)

SoftUni Foundation

- JavaScript (JS) is untyped language
  - Untyped (dynamically typed) == variables have no types
  - Data (values) still have a type





Variables and Operators in JS

SoftUni Foundation

- Define variables with the keyword `let`

```
let i = 5; let j = 3; // i and j are numbers
console.log(typeof(i)) // Number
i = i * j; i++; // 16
console.log(i + 1); // 17
```

The ; can be omitted

```
let str = 'Hello'; // string
str = str + " JS";
console.log(str); // Hello JS
```

Strings are in format '…' or "…"

```
console.log(s2); // Uncaught ReferenceError: s2 is not defined
```

Undefined identifier usage

Problem: Calculate Expression

SoftUni Foundation

- Write a JavaScript program to print the value of the following expression: **(30 + 25) + ((35 – 14) * 2)**
- Sample solution:

```
function solve() {
      let val = (30 + 25) + ((35 – 14) * 2);
      console.log(val);
}
```

Type Conversions

SoftUni Foundation

- All JS types can be compared due to automatic type conversion

```
5 == "5.0" // true (== checks value)
5 === "5.0" // false (=== checks type + value)
1 == true // true
[] == false // true
'' == false // true
"" == 0 // true
5 + false + true === 6 // true
```

- More examples to play with:
  - WAT, JS Weird Parts

Conditions: if-else

- JavaScript implements the classical `if` / `if-else` statements:

```
let number = 5;


if (number % 2 == 0) {
  console.log("Even number");
} else {
  console.log("Odd number");
}
```

Problem: Bigger Number

SoftUni Foundation

- Write a JavaScript program to print bigger of two numbers:
  - **let firstNumber;**
  - **let secondNumber;**
- Sample solution:

```
let firstNumber = 100;
let secondNumber = 200;
```

Solution: Bigger Number

SoftUni Foundation

```
function solve(input) {
        let firstNumber = Number(input[0]);
        let secondNumber = Number(input[1]);

        if (firstNumber > secondNumber) {
                console.log(firstNumber);
        } else {
                console.log(secondNumber);
        }
}
```

The switch-case Statement

SoftUni Foundation

- Selects a statement from a list depending on the value of the **switch** expression

```
let day = 3
switch (day) {
   case 1: console.log('Monday'); break;
   case 2: console.log('Tuesday'); break;
   case 3: console.log('Wednesday'); break;
    …
   case 7: console.log('Sunday'); break;
   default: console.log('Error!'); break;
}
```

Loops: for, while, do-while, …

SoftUni Foundation

- The `for` / `while` / `do-while` loops work as in C# and Java

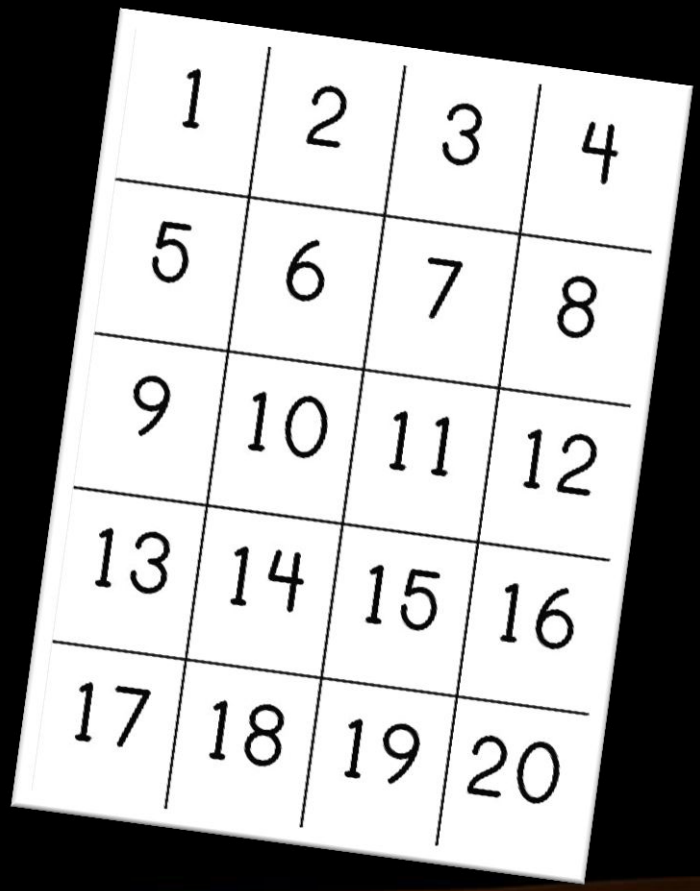
```
for (let i = 0; i <= 10; i++)
  console.log(i); // 0 1 2 3 4 … 10
```

```
let count = 1;
while (count < 1024)
  console.log(count *= 2); // 2 4 8 16 … 1024
```

```
let s = "ha“;
do { console.log(s); s = s + s; }
while (s.length < 10); // ha haha hahahaha
```

Problem: Print first 20 numbers

- Write a JavaScript program to print first 20 numbers:
- Use for, while or do-while loop:

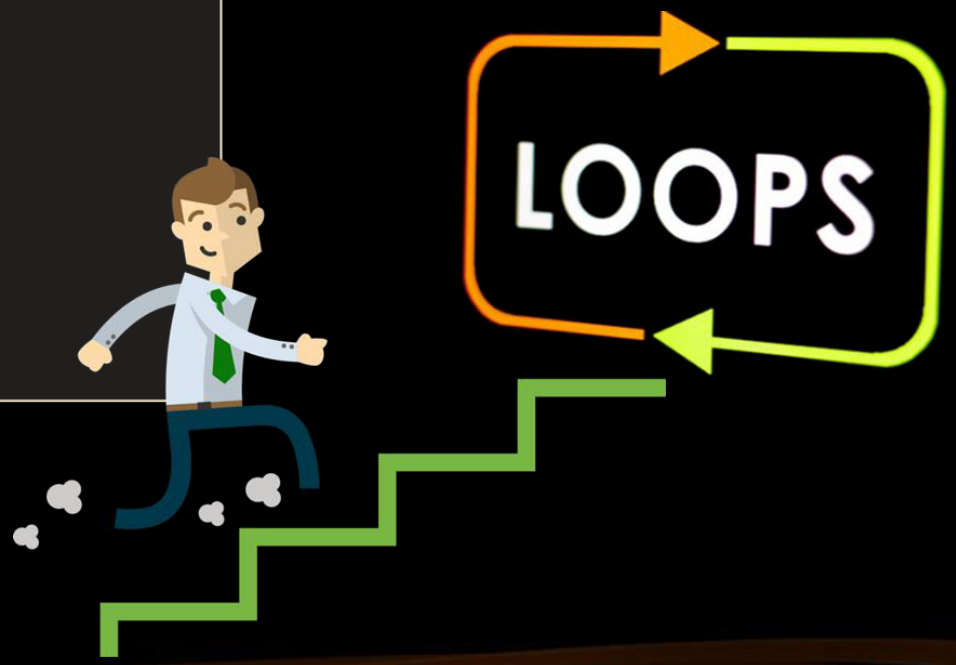


Solution: First 20 Numbers

SoftUni Foundation

It's so simple:

```
function solve() {
   for (let i = 1; i <= 20; i++) {
       console.log(i);
   }
}
```



Functions

SoftUni Foundation

- Functions in JS hold a piece of code (script)
  - Can take parameters and return result
  - Similar to functions in C and PHP and methods in C++ / C# / Java

```
function multiply(a, b) {
  return a * b;
}
console.log(multiply(2, 3)); // 6 == 2 * 3
console.log(multiply(2)); // NaN == 2 * undefined
console.log(multiply(5, 6, 7)); // 30 == 5 * 6
```

Anonymous Functions and Callbacks

SoftUni Foundation

```
let print = function(x) { console.log(x) };
[10, 20, 30].forEach(print) // 10 20 30
```

```
let sum = 0;
[10, 20, 30].forEach(function(x) {
  sum += x;
});
console.log(sum) // 60
```

```
(function(text) { alert(text) })("hello")
```

Block Scope vs. Function Scope

SoftUni Foundation

- **var** defines "function scope"

```
(function functionScope() {
  console.log(x); // undefined
  for (var x = 1; x < 10; x++)
    console.log(x);; // 1 … 9
  console.log(x); // 10
  var x = 5; console.log(x); // 5
  var x = 2; console.log(x); // 2
  x = 3; console.log(x) // 3
}) ();
```

- Use **var** carefully!

Block Scope vs. Function Scope (2)

SoftUni Foundation

- `let` defines "block scope"

```
(function blockScope() {
  // console.log(x); // error!
  let x = 5; console.log(x); // 5
  for (let x = 20; x < 30; x++)
    console.log(x); // 20 … 29
  console.log(x); // 5
  x = 10; console.log(x); // 10
  // let x = 5; // error!
}) ();
```

- Prefer **let** in most cases

Objects

SoftUni Foundation

- Objects in JavaScript hold key-value pairs:

```
let obj = { name : "SoftUni", age : 2 };
console.log(obj); // Object {name: "SoftUni", age: 2}
obj['site'] = "http://www.softuni.bg";
obj.age = 10;
obj['name'] = "Software University";
console.log(obj); // Object {name: "Software
University", age: 10, site: "http://www.softuni.bg"}
delete obj.name;
delete obj.site;
console.log(obj); // Object {age: 10}
```

Objects and JSON

- JavaScript objects can be stored as text in JSON format

```
let obj = { name : "SoftUni", age : 2 }
let str = JSON.stringify(obj);
console.log(str);
// { "name":"SoftUni", "age":2 }
```


```
let str = "{\"name\":\"Nakov\",\"age\":24}"
let obj = JSON.parse(str);
console.log(obj); //
Object { name: "Nakov", age: 24 }
```

Arrays in JavaScript

SoftUni Foundation

- Arrays in JS can hold mixed types:

```
// Array holding numbers
let numbers = [1, 2, 3, 4, 5];

// Array holding strings
let weekDays = ['Monday', 'Tuesday', 'Wednesday',
  'Thursday', 'Friday', 'Saturday', 'Sunday'];

// Array of mixed data
var mixedArr = [1, new Date(), 'hello'];
```

Problem: Print the elements in array

- Write a JavaScript program to print all of the elements in array:
- Create random array like this:
- Put your solution in function

```
let numbers = [1, 2, 3, 4, 5];
```

Solution: Print the elements in array

SoftUni Foundation

```
function solve(input) {
      for (let element of array) {
            console.log(element);
      }
}
```

```
function solve(input) {
      for (let index in array) {
            console.log(array[index]);
      }
}
```

Processing Arrays Elements

- Print all elements of an array of strings:

```
let capitals = ['Sofia', 'Washington', 'London',
'Paris'];

for (let capital of capitals)
  console.log(capital);

for (let i in capitals)
  console.log(capitals[i]);

for (let i = 0; i < capitals.length; i++)
  console.log(capitals[i]);
```

Works like **foreach**

This is not **foreach**! It goes through the array indices.

Traditional **for**-loop

Array Operations

SoftUni Foundation

```
let numbers = [1, 2, 3, 4];
console.log(numbers.join('|')); // result: 1|2|3|4|5

numbers.push(5);
console.log(numbers.join('|')); // result: 1|2|3|4|5

let tail = numbers.pop();       // tail = 5;
console.log(numbers.join('|')); // result: 1|2|3|4

numbers.unshift(0);
console.log(numbers.join('|')); // result: 0|1|2|3|4

let head = numbers.shift();     // head = 0;
console.log(numbers.join('|')); // result: 1|2|3|4
```

Strings

- Strings in JavaScript hold a sequence of Unicode characters

```
let str1 = "Some text in a string variable";
let str2 = 'Text enclosed in single quotes';
for (let i = 0; i < str1.length; i++)
  console.log(str1[i] + ' ' + str2[i]);
```

```
let tokens = 'C#, Java, PHP ,HTML'.split(',');
// tokens = ['C#', ' Java', ' PHP ', 'HTML']
tokens = tokens.map(s => s.trim());
console.log(tokens);
// ['C#', 'Java', 'PHP', 'HTML']
```

Map by lambda function

Summary

- JavaScript is a dynamically-typed language
  - Commands are arranged in scripts
- Program logic (variables, conditions, loops) are similar to C# / Java / PHP / C++
- Arrays in JS combine traditional arrays, lists and dictionaries
- Objects in JS hold key-value pairs
- JS is a functional language: relies on functions, callbacks, lambdas, etc.

Web Fundamentals – Course Introduction

SoftUni Foundation

Questions?

https://softuni.bg/courses/web-fundamentals-html5

SoftUni Diamond Partners

SoftUni Foundation

INDEAVR
Serving the high achievers

INFRAGISTICS®

SoftwareGroup
doing it right

xssoftware

NETPEAK
SEO and PPC for Business

SUPER HOSTING .BG®

SoftUni Diamond Partners

SoftUni Foundation

License

SoftUni Foundation

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg
- Software University Foundation
  - http://softuni.foundation/
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

Software University

SoftUni Foundation

FORUM