



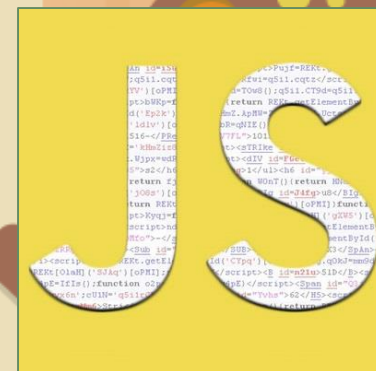
WEB ACADEMY

Front End за Начинаещи JavaScript



JavaScript

```
<script>  
  document.write('Awesome!');  
</script>
```



За JavaScript

5

- JavaScript – интерпретаторен език за програмиране
- Създаден 4.12.1995 година от Netscape
- Изпълнява се в браузъра
- Езикът е чувствителен към регистъра
- Поддържа функционален и обектно ориентиран стил
- Няма нищо общо с Java освен приликата в името
- Поддържа едноредови коментари във формат
// Това е едноредов коментар и няма да се интерпретира от браузъра
- Поддържа многоредови коментари във формат
/* Това е многоредов коментар, който няма да се интерпретира от браузъра */
- Няколко оператора се групират посредством { и }



ВЪЗМОЖНОСТИ

6

- **Какви са възможностите на JavaScript:**
 - Зареждане на данни чрез AJAX
 - Ефекти с изображения и HTML елементи
 - Управление на прозорци и рамки.
 - Разпознаване на възможностите на браузъра.
 - Използване на камерата и микрофона.
 - Създаване на графики
- **Какво не може да прави JavaScript:**
 - Не може да записва информация на потребителския компютър
 - Не може да записва информация на отдалечения сървър
 - Не може да запазва информация директно в база данни.
 - Не може да се стартират локални приложения.



Вмъкване на JavaScript



- чрез деклариране на външен файл вътре в `<head>` тага
`<script src="file.js"></script>`
- чрез използване на `<script>` таг
`<script>`
`<!--`
 `alert('Здравейте');`
`-->`
`</script>`
- чрез директно задаване на JS код на ниво елемент
`<i onclick="alert('Здрасти');">Кликнете тук</i>`


Добрата практика изисква:

1. Функциите да се декларират чрез метод 1 или 2
2. На ниво елемент, да се прави извикване на вече дефинираните функции



Променливи



- Няколко думи за променливите:
 - Променливите са динамични и слаботипизирани
 - Декларират се чрез ключовата дума `var` следвана от името
 - Името на променливата може да съдържа букви, цифри, "_" и "\$" 
 - Името на променливата не може да започва с цифра
 - Името на променливата не може да бъде ключова дума или оператор
 - Възможно е присвояване на стойност, заедно с декларацията
 - По подразбиране всяка променлива има стойност `undefined`
- Ами константите? - JavaScript няма константи!

//Валидни имена на променливи: ninja, apples_and_oranges, var56
//Невалидни имена на променливи: 3, var, "test", if, while, 5wtx1

```
var firstVar; // Деклариране на променлива с име firstVar  
var secondVar = 1 // Деклариране и инициализиране на стойност
```



JavaScript Keywords



abstract	arguments	boolean	break	byte
case	catch	char	class*	const
continue	debugger	default	delete	do
double	else	enum*	eval	export*
extends*	false	final	finally	float
for	function	goto	if	implements
import*	in	instanceof	int	interface
let	long	native	new	null
package	private	protected	public	return
short	static	super*	switch	synchronized
this	throw	throws	transient	true
try	typeof	var	void	volatile
while	with	yield	<i><u>Виж още: пълен списък</u></i>	



Типове данни



Типове данни поддържани от езика:

- числа - например 1, 2, 3, -3.14 и други
- низове - задават се в "кавички" или 'апострофи'
- логически (булеви) - true или false
- масиви
 - задават се чрез []
 - съдържат списък с елементи
 - елементите могат да бъдат от различни типове
 - възможно е да има вложени масиви
 - ключовете на масивите са задължително цели числа, в диапазона от 0..N-1, където N е броя на елементите

Типове данни

11

/ Примери за типове данни */*

var a = 3; *//Инициализиране на числова променлива:*

var b = "JavaScript iz kuwl"; *//Инициализиране на променлива с низ*

var c = **true**; *//Инициализиране на булева променлива*

//създаване на нов масив:

var empty = []; *//празен масив*

var my_data = [1, 2, 3, "hi", "bye", -2.11];

var array_of_arrays = [[1, 2, 3], [4, 5, 6], "anything else?"]; *//масив от масиви*

//Извеждане на типа на данните в променливите

alert(**typeof** a);

alert(**typeof** b);

alert(**typeof** c);



Camel Case



Historically, programmers have used three ways of joining multiple words into one variable name:

Hyphens:

first-name, last-name, master-card, inter-city.

Underscore:

first_name, last_name, master_card, inter_city.

Camel Case:

FirstName, LastName, MasterCard, InterCity.



In programming languages, especially in JavaScript, camel case often starts with a lowercase letter:

firstName, lastName, masterCard, interCity.



Въвеждане на данни



В JavaScript може да въвеждаме данни по няколко начина:

- чрез диалогов прозорец - `prompt(text, default_text);`
- чрез диалог за потвърждение - `confirm(text);`
- чрез взимане стойността на HTML елемент

```
<html>
<body>
  <input type="text" name="user_name" id="user_name" />
  <script>
    var confirm_value = confirm('Are you sure?');
    var prompt_value = prompt("Please, enter text", "Default Text...");
    var user_name_value = document.getElementById("user_name").value;
  </script>
</body>
</html>
```

- Виж още: [JavaScript HTML Input Examples](#)



Извеждане на данни



JavaScript може да извежда данни по няколко начина:

- чрез системно съобщение - `window.alert()`;
- вътре в HTML документ - `document.write()`;
- вътре в HTML елемент - `innerHTML`
- в конзолата на браузъра - `console.log()`

```
<html>
<body>
  <script>
    window.alert(1 + 2);
    document.write(3 + 4);
    document.getElementById("demo").innerHTML = 5 + 6;
  </script>
  <button onclick="document.write(7 + 8)">Try it</button>
</body>
</html>
```



Оператори



JavaScript работи със следните оператори:

- оператор за присвояване =
- празен оператор ;
- аритметични оператори

- събиране +
- изваждане -
- умножение *
- деление /
- деление по модул %
- увеличаване с 1 ++
- намаляване с 1 --

- комбинирани оператори: += , -= , *=, /=
- логически оператори: <, >, <=, >=, ==, !=, ===, !==, &&, ||

Примери:

```
var x = 5;      // assign the value 5 to x
var y = 2;      // assign the value 2 to y
var z = x + y;  // assign the value 7 to z
console.log(x % y);
```

```
txt1 = "John";
txt2 = "Doe";
txt3 = txt1 + " " + txt2;
```

```
x = 5 + 5;
y = "5" + 5;
z = "Hello" + 5;
```



Проверка на условие



В JavaScript има 2 основни начина за проверка на условие:

- if оператор

- синтаксис: `if (условие) { оператори; }else{ оператори; }`
- ако условието е истина, се изпълняват операторите
- ако условието е лъжа, се изпълняват `else операторите`
- скобите `{` и `}` са важни!

- троен оператор

- синтаксис: `(условие) ? оператори; : оператори;`
- ако условието е истина, се изпълняват операторите
- ако условието е лъжа, се изпълняват операторите

Проверка на условие



```
<html>
<body>
<p>Въведете години:</p><input id="age" value="18" />
<button onclick="checkMe()">Try it</button><p id="result"></p>
<script>
function checkMe() {
    var age, voteable;
    age = Number(document.getElementById("age").value);
    if (isNaN(age)) {
        voteable = "Грешни входни данни!";
    } else {
        voteable = (age < 18) ? "Много сте млад!" : "Добре влизай!";
    }
    document.getElementById("result").innerHTML = voteable;
}
</script>
</body>
</html>
```

Избор на вариант



switch конструкция

- Позволява избор на вариант измежду няколко
- Проверява за стойност и тип!!!
- Не забравяйте **break; !!!**
- Синтаксис:

```
switch(израз) {  
    case n:  
        code block  
        break;  
    case n:  
        code block  
        break;  
    default:  
        default code block  
}
```

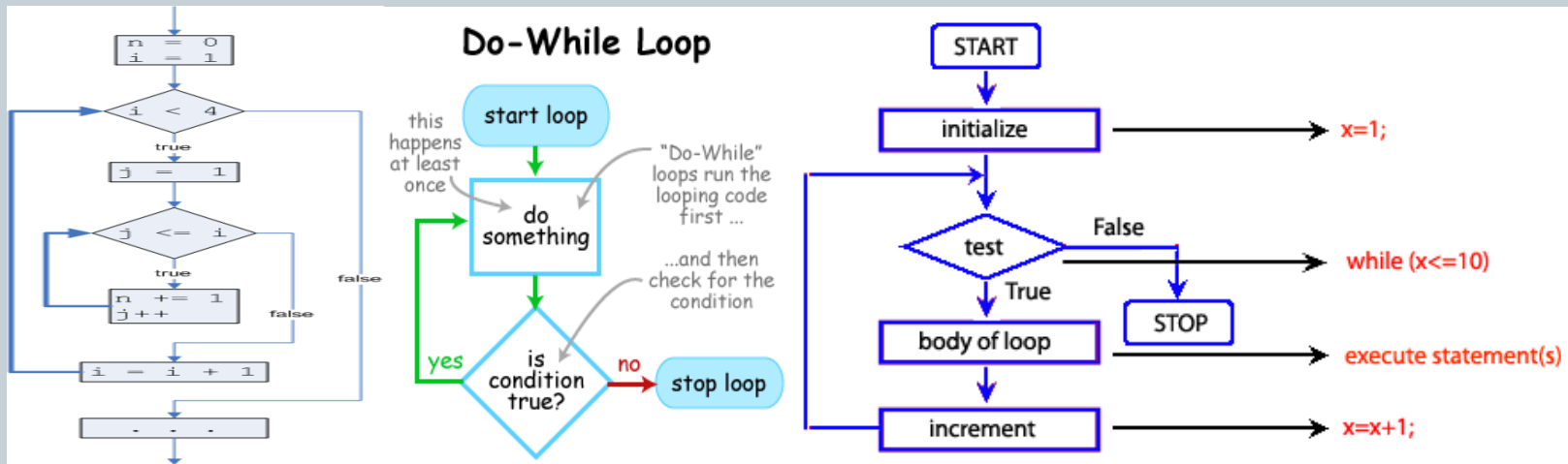
Пример:

```
switch (new Date().getDay()) {  
    case 0:  
        day = "Неделя";  
        break;  
    case 1:  
        day = "Понеделник";  
        break;  
    case 2:  
        day = "Вторник";  
        break;  
    default:  
        day = "Непознат ден";  
}
```


Цикли



- оператор for - изпълнява се определен брой пъти
- оператор while - изпълнява се 0 или повече пъти
- оператор do..while - изпълнява се най-малко 1 път




Още по темата:


- http://www.w3schools.com/js/js_loop_for.asp
- http://www.w3schools.com/js/js_loop_while.asp

Често Срещани Грешки

20

-  Случайно използване на оператор за присвояване (=), вместо оператор за сравнение (==) може да доведе до неочаквани резултати!


```
var x = 0;  
if (x == 10){} // Коректно използване на оператор за сравнение  
if (x = 10){} // Връща true, защото присвоява на X стойност 10, а 10 > 0  
if (x = 0){} // Връща false, защото присвоява на X, стойност 0, а 0 = false
```

-  Случайно използване на оператор за сравнение (==), вместо оператор за идентичност (===)

```
var x = 10;  
var y = "10";  
if (x == y){} // Връща true, защото сравнява само стойността  
if (x === y){} // Връща false, защото сравнява стойността и типа
```

Често Срещани Грешки

21

 Грешно предположение, че конструкцията switch...case проверя за равни стойности, вместо за идентичност

// Запомнете: switch конструкцията проверява за идентичност, т.е.

```
var x = 10;  
switch(x) {  
    case 10: alert("Число"); break;  
    case "10": alert("Низ"); break;  
}
```

// Ще изведе съобщение за Число, защото проверява и стойност и тип

 Искате събиране, а се получава слепване (конкатенация)

// Оценяването на аргументите става от ляво надясно!

```
var x = 10 + 5;           // резултата ще бъде 15  
var x = 10 + "5";         // резултата ще бъде "105"  
var x = 10 + 6 + "29";    // резултата ще бъде "1629"
```

Често Срещани Грешки

22

- Неразбиране на това, как работят float числата

// Запомнете: бъдете внимателни при работа с числа с плаваща запетая

```
var x = 0.1;
```

```
var y = 0.2;
```

```
var z = x + y      // очаквате, че стойността на Z е 0.3, но грешите!!!
```

```
if (z == 0.3)      // връща false, защото стойността на Z е 0.3000000004
```

// За да решите проблема, използвайте следния малък трик :-)

```
var z = (x * 10 + y * 10) / 10;    // z will be 0.3
```

Грешно прекъснат низов литерал

```
var x =              // Работи коректно
```

```
"Hello World!";
```

```
var x = "Hello      // НЕ работи, ще изведе синтактична грешка!  
World!";
```


```
var x = "Hello \     // Работи коректно.
```

```
World!";            // Бъдете внимателни с този синтаксис! :)
```



Често Срещани Грешки

23

 Грешно предположение, че масивите са асоциативни. В много програмни езици масивите са асоциативни, но тук НЕ са, т.е. техните ключове винаги са цели числа

```
var person = [];  
person[0] = "John";  
person[1] = "Doe";  
person[2] = 46;  
var x = person.length; // връща 3  
var y = person[0];      // "John"  
// Коректен код
```

```
var person = [];  
person["firstName"] = "John";  
person["lastName"] = "Doe";  
person["age"] = 46;  
var x = person.length; // връща 0  
var y = person[0];      // undefined  
// НЕкоректен код
```

 Завършване на дефиницията на масив със запетая (,)

```
var points = [40, 100, 1, 5, 25, 10,]; // НЕкоректен код. НЕ пишете така!  
var points = [40, 100, 1, 5, 25, 10]; // Коректен код. Пишете го така!
```



Често Срещани Грешки

24

 Други трудно откриваеми грешки:

- Грешно поставен оператор ;
- Изпуснат оператор ;
- Символ изписан на кирилица, вместо на латиница
- Липсваща или неправилно поставена { скоба }
- Неразбиране на разликата между масиви и обекти
- Неразбиране на разликата между `undefined` и `null`.
`null` е за обекти, `undefined` е за променливи и методи.
- Семантични грешки, се откриват изключително трудно, след дълбоко и подробно тестване.
- Повече информация: [тук](#)

ВЪПРОСИ

58

