



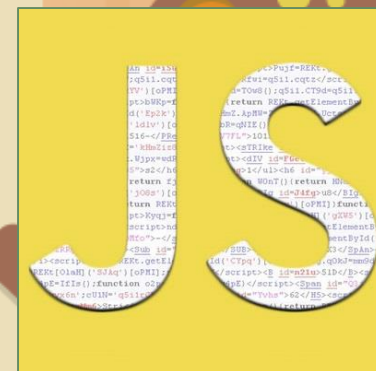
WEB ACADEMY

Front End за Начинаещи JavaScript




JavaScript

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




Често Срещани Грешки

5

-  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

6

 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

7

- 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

8

 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

9

 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](#)

Функции



- Функциите представляват самостоятелни парчета програмен код, които решават определена задача.
- Всяка функция се състои от име, списък с формални (входни) параметри, тяло и изходни параметри.
- Параметрите се разделят помежду си със запетая ","
- Функцията се дефинира с формални параметри, и се извиква (инвокира) с реални аргументи
- Функция, която не връща стойност се нарича още void функция, а в някои езици се нарича също така процедура

- Синтаксис:

```
function toCelsius(fahrenheit) {  
    return (5/9) * (fahrenheit-32);  
}  
document.getElementById("d").innerHTML = toCelsius(77);
```



Функции



Защо да използваме функции:

- кодът е преизползваем
- пишете веднъж използвате многократно
- различни входни параметри дават различен резултат
- ако е нужна промяна, променяте само на едно място

Задачи за самоподготовка върху функции:

- извикайте функцията
- поправете функцията
- допишете тялото на функцията
- извикайте функцията с параметри
- дефинирайте функцията myFunction, която извежда текст

Повече информация: [тук](#)



```
var jsObject = {  
  name: "Caitlyn",  
  age: 24,  
  favorite_foods: ['pizza', 'subway sandwiches', 'soup'],  
  quirk: "I hate bunched up seeds. They look like larvae."  
};
```

Обекти (Въведение)



- Обектите представляват съвкупност от свойства и методи
- Свойствата представляват променливи за обекта
- Методите представляват функции за обекта
- JS обектите са асоциативни (за разлика от масивите)

Object	Properties	Methods
	<p>car.name = Fiat</p> <p>car.model = 500</p> <p>car.weight = 850kg</p> <p>car.color = white</p>	<p>car.start()</p> <p>car.drive()</p> <p>car.brake()</p> <p>car.stop()</p>

Пример за обект кола (car)

```
var jsObject = {  
  name: "Caitlyn",  
  age: 24,  
  favorite_foods: ['pizza', 'subway sandwiches', 'soup'],  
  quirk: "I hate bunched up seeds. They look like larvae."  
};
```

Обекти (Синтаксис)



- Как се дефинира обект в JavaScript:

- името на свойството / метода НЕ се загражда в кавички / апострофи
- отделните свойства / методи се разделят помежду си със , (запетая)
- между името на свойството / метода и неговата стойност се слага :
- цялата дефиниция на обекта се загражда във { } (фигурни скоби)
- след затварящата } (фигурна скоба) се слага ; (точка и запетая)
- можете да се обръщате към свойствата на обекта по 2 начина:
objectName.propertyName или *objectName["propertyName"]*
- **НЕ** създавайте инстанции на обектите String, Number, Boolean, тъй като това ще се отрази негативно върху бързодействието

// Това може да бъде потенциален проблем. НЕ пишете така!

```
var x = "John";  
var y = new String("John");
```

// (x === y) е false, x и y имат различни типове (string и object)

```
var jsObject = {  
  name: "Caitlyn",  
  age: 24,  
  favorite_foods: ['pizza', 'subway sandwiches', 'soup'],  
  quirk: "I hate bunched up seeds. They look like larvae."  
};
```

Обекти (Примери)



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

<script>

var d = new Date(); // Date е вграден в езика обект. Очаквайте повече скоро

var person = {

firstName: "Yordan",

lastName: "Enev",

age : 29,

favmovies: ['The Lord of the Rings', 'The Hobbit', 'The Notebook'],

fullName: function() {

return this.firstName + " " + this.lastName;

}

};

document.getElementById("demo").innerHTML = person.fullName();

alert('Рождена година: ' + (d.getFullYear() - person.age));

</script>



Методи за работа с данни



JavaScript има много и различни методи за работа с данни. Най-общо те могат да бъдат групирани в няколко обекта:

- Math обект - за работа с математически операции
- Date обект - за работа с дати
- Array обект - за работа с масиви
- String обект - за работа с низове
- RegExp обект - за работа с регулярни изрази

Внимание: Не създавайте прекалено много обекти, защото това се отразява зле на бързодействието и може да доведе до неочаквани последици...

Math обект



Math обектът ви позволява да работите с аритметични операции и методи. Той няма конструктор, и всички обръщания стават през него, без да се създава явно негова инстанция.

Property	Description	Константи на Math обекта
<u>E</u>	Returns Euler's number (approx. 2.718)	
<u>LN2</u>	Returns the natural logarithm of 2 (approx. 0.693)	
<u>LN10</u>	Returns the natural logarithm of 10 (approx. 2.302)	
<u>LOG2E</u>	Returns the base-2 logarithm of E (approx. 1.442)	
<u>LOG10E</u>	Returns the base-10 logarithm of E (approx. 0.434)	
<u>PI</u>	Returns PI (approx. 3.14)	
<u>SQRT1_2</u>	Returns the square root of 1/2 (approx. 0.707)	
<u>SQRT2</u>	Returns the square root of 2 (approx. 1.414)	



Методи на Math обекта

Method	Description
<code>abs(x)</code>	Returns the absolute value of x
<code>acos(x)</code>	Returns the arccosine of x, in radians
<code>asin(x)</code>	Returns the arcsine of x, in radians
<code>atan(x)</code>	Returns the arctangent of x as a numeric value between $-\pi/2$ and $\pi/2$ radians
<code>atan2(y,x)</code>	Returns the arctangent of the quotient of its arguments
<code>ceil(x)</code>	Returns the value of x rounded up to its nearest integer
<code>cos(x)</code>	Returns the cosine of x (x is in radians)
<code>exp(x)</code>	Returns the value of E^x
<code>floor(x)</code>	Returns the value of x rounded down to its nearest integer
<code>log(x)</code>	Returns the natural logarithm (base E) of x
<code>max(x,y,z,...,n)</code>	Returns the number with the highest value
<code>min(x,y,z,...,n)</code>	Returns the number with the lowest value
<code>pow(x,y)</code>	Returns the value of x to the power of y
<code>random()</code>	Returns a random number between 0 and 1
<code>round(x)</code>	Returns the value of x rounded to its nearest integer
<code>sin(x)</code>	Returns the sine of x (x is in radians)
<code>sqrt(x)</code>	Returns the square root of x
<code>tan(x)</code>	Returns the tangent of an angle



Примери за работа с Math

18

Опитайте следните примери:

Math.PI;	// връща стойността на константата Pi
Math.round(4.7);	// връща 5
Math.round(4.4);	// връща 4
Math.pow(8,2);	// връща 64
Math.sqrt(64);	// връща 8
Math.abs(-4.7);	// връща 4.7
Math.ceil(4.4);	// връща 5 закръгля нагоре до цяло число
Math.floor(4.7);	// връща 4 закръгля надолу до цяло число
Math.sin(90 * Math.PI / 180);	// връща 1 (синус на 90 градуса)
Math.cos(0 * Math.PI / 180);	// връща 1 (косинус на 0 градуса)
Math.min(0, 150, 30, 20, 8, -200);	// връща -200
Math.max(0, 150, 30, 20, 8, -200);	// връща 150
Math.random();	// връща случайна стойност между [0,1) !!!

Още по темата: [тук](#)



Случайни числа с Math

19

Опитайте следните примери:

```
Math.random();           // връща случайна стойност между [0,1)
Math.floor(Math.random() * 10); // връща случайно число между 0 и 9
Math.floor(Math.random() * 11); // връща случайно число между 0 и 10
Math.floor(Math.random() * 10) + 1; // връща случайно число между 1 и 10
```

```
// Функцията връща случайно число в интервала [min, max)
function getRndInteger(min, max) {
    return Math.floor(Math.random() * (max - min) ) + min;
}
```

```
// Функцията връща случайно число в интервала [min, max]
function getRndInteger(min, max) {
    return Math.floor(Math.random() * (max - min + 1) ) + min;
}
```

Още по темата: [тук](#)



Date обект



Date обектът ви позволява да работите с дати.

Има 4 начина за инициализация на Date обекта:

- new Date()
- new Date(milliseconds)
- new Date(dateString)
- new Date(year, month, day, hours, minutes, seconds, milliseconds)

Датата може да се задава като низ, в следните 4 формата:

Type	Example
ISO Date	"2015-03-25" (The International Standard)
Short Date	"03/25/2015" or "2015/03/25"
Long Date	"Mar 25 2015" or "25 Mar 2015"
Full Date	"Wednesday March 25 2015"



Методи на Date обекта



Често използвани методи за взимане на дата:

Method	Description
getDate()	Get the day as a number (1-31)
getDay()	Get the weekday as a number (0-6)
getFullYear()	Get the four digit year (yyyy)
getHours()	Get the hour (0-23)
getMilliseconds()	Get the milliseconds (0-999)
getMinutes()	Get the minutes (0-59)
getMonth()	Get the month (0-11)
getSeconds()	Get the seconds (0-59)
getTime()	Get the time (milliseconds since January 1, 1970)

Методи на Date обекта



Често използвани методи за задаване на дата:

Method	Description
setDate()	Set the day as a number (1-31)
setFullYear()	Set the year (optionally month and day)
setHours()	Set the hour (0-23)
setMilliseconds()	Set the milliseconds (0-999)
setMinutes()	Set the minutes (0-59)
setMonth()	Set the month (0-11)
setSeconds()	Set the seconds (0-59)
setTime()	Set the time (milliseconds since January 1, 1970)

Вижте още: [пълен списък с методи на Date обекта](#)



Събития



- Събитията (events) възникват, когато нещо се случи, а JavaScript от своя страна реагира на тези събития
- Свързват се с конкретен елемент от страницата

```
<button onclick="this.innerHTML=Date()">The time is?</button>
```

- Най-често използвани събития:

Event	Description
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

