

1. History

- JavaScript was invented by Brendan Eich in 1995
- First named as LiveScript 1995
- Jscript by Microsoft in 1996
- Became an ECMA standard in 1997
- ECMAScript 6 (released in June 2015) is the latest official version of JavaScript

2. Introduction

JavaScript is a cross-platform

object-oriented scripting language

small and lightweight language

Client-side JavaScript extends the core language by supplying objects to control a browser and its Document Object Model (DOM).

Server-side JavaScript extends the core language by supplying objects relevant to running JavaScript on a server. For example, server-side extensions allow an application to communicate with a database, provide continuity of information from one invocation to another of the application, or perform file manipulations on a server.

3. Key Ideas

Load and Go delivery

Loose typing

Objects as general containers

Linkage through Global Variables

4. Script

Inline JavaScript

External JavaScript

Where to place js file?

Async attribute

```
<script src="demo_async.js" async></script>
```

If async is present: The script is executed asynchronously with the rest of the page (the script will be executed while the page continues the parsing)

If async is not present and defer is present: The script is executed when the page has finished parsing

If neither async or defer is present: The script is fetched and executed immediately, before the browser continues parsing the page

5. Data Types

The latest ECMAScript standard defines seven data types:

Boolean. true and false.

null. A special keyword denoting a null value.

undefined. A top-level property whose value is undefined.

Number. 42 or 3.14159.

String. "Howdy"

Object

Symbol (new in ECMAScript 6). A data type whose instances are unique and immutable

6. Numbers

Only One number type(no =>int, float)

64-bit floating point

IEEE-754

Does not map well to common understanding of arithmetic.

$0.1 + 0.2 = 0.30000000000000004$

NaN(special number)

Toxic: Any arithmetic operation with NaN as input will have NaN as result.

NaN is not equal to anything, including NaN

`typeof(NaN)=?`

`Numbers(value)`

Converts string into no.

`parseInt(value , radix)`

It stops at the first non-digit character.

`parseFloat(value)`

Parses a string and returns a floating point number.

Math

The Math object allows you to perform mathematical tasks.

8. Strings

Sequence of 0 or more 16-bit characters

UCS-2,not quite UTF-16

No separate character type

String literal can use single or double quotes

Strings are immutable

String methods

<code>charAt</code>	<code>concat</code>
<code>indexOf</code>	<code>split</code>
<code>replace</code>	<code>substring</code>

`toLowerCase` `toUpperCase`

9. JavaScript Output

`window.alert()`

Writing into an alert box

`document.write()`

Writing into the HTML

The `document.write()` method should be used only for testing.

`innerHTML`

To access an HTML element,

`document.getElementById("id").innerHTML = "hello";`

`console.log()`

10. JavaScript Variable Scope

Global Variables – A global variable has global scope which means it can be defined anywhere in your JavaScript code.

Local Variables – A local variable will be visible only within a function where it is defined. Function parameters are always local to that function.

`var x=6;`

`num=9;`

`let num=10;`

Variable hoisting

JavaScript only hoists declarations, not initializations.

11. Strict Mode

"use strict"; Defines that JavaScript code should be executed in "strict mode"

Strict mode is supported in:

Internet Explorer from version 10. Firefox from version 4.

Chrome from version 13. Safari from version 5.1. Opera from version 12.

Strict mode makes it easier to write "secure" JavaScript.

Strict mode changes previously accepted "bad syntax" into real errors

ex:

```
"use strict";  
x = 3.14;           // This will cause an error (if x has not been declared)
```

12. Falsy Values

false

Null

Undefined

"" (empty string)

0

NaN

All other values (including all objects) are truthy.

13. Object

JavaScript objects are containers for named values.

Objects in JavaScript are mutable keyed collections

`new Object()` produces an empty container of name/value pairs.

In JavaScript, arrays are objects, functions are objects, regular expressions are objects, and, of course, objects are objects.

A name can be any string, a value can be any value except undefined.

```
var person = {firstName : "Michal", lastName : "jackson"};
```

Members can be accessed by dot notation or subscript notation

```
person.lastName OR person["lastName"]
```

14. Object

A property name can be any string, including the empty string.

A property value can be any JavaScript value except for undefined.

The new operator is used to create an instance of an object.

```
var employee = new Object();  
var books = new Array("C++", "Perl", "Java");  
var day = new Date("August 15, 1947");
```

15. Arrays

The Array object lets you store multiple values in a single variable.

```
var fruits = new Array( "apple", "orange", "mango" );  
var fruits = [ "apple", "orange", "mango" ];
```

For simplicity, readability and execution speed, use the second one (the array literal method).

Adding Array Elements

```
fruits.push("Lemon");
```

Access the Elements of an Array

```
fruits[0];
```

JavaScript does not support arrays with named indexes.

16. Arrays cont

You Can Have Different Objects in One Array

```
myArray[0] = Date.now;  
myArray[1] = myFunction;  
myArray[2] = fruits;
```

Multi-Dimensional Array

```
var items = [[1,2],[3,4],[5,6]];  
alert(items[0][0]);
```

17. Arrays Methods

pop()

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

push()

```
fruits.push("Kiwi"); //adds a new element to an array (at the end):
```

shift()

```
fruits.shift();
```

```
//removes first element of an array, and "shifts" all other elements one place up.
```

unshift()

```
fruits.unshift("Lemon");
```

```
//adds a new element to an array (at the beginning), and "unshifts" older elements
```

18.

splice()

```
fruits.splice(0, 1);
```

```
//to remove elements without leaving "holes" in the array
```

slice()

```
var citrus = fruits.slice(1, 3);
```

```
//slices out a piece of an array into a new array:
```

sort() & reverse()

```
fruits.sort();      // Sorts the elements of fruits
```

```
fruits.reverse();   // Reverses the order of the elements
```

19. Operators

== and !=

- These operators can do type coercion

`("5"==5)`

`("5"===5)`

- Its better to use `===` and `!==`, which do not do type coercion.
- `||` operator

if first operand is truthy then result is first else result is second

it can be used to fill default values

`var last=input || otherVal;`

20. Operators

- `&&`(logical AND)

If the first operand is truthy

then result is second operand

else result is first operand


- Bitwise

`& | ^ >> >>> <<`

Bitwise operators convert the operand to a 32 bit signed integer, and turn into 64-bit floating point no.

the result back

21. Operator precedence

Value	Operator	Description	Example
19		Expression grouping	(3 + 4)
18	.	Member	person.name
18	[]	Member	person["name"]
17	()	Function call	myFunction()
17	new	Create	new Date()
16	++	Postfix Increment	++i
16	--	Postfix Decrement	--i
15	++	Prefix Increment	i++
15	--	Prefix Decrement	i--
15	!	Logical not	!(x==y)
15	typeof	Type	typeof x
14	*	Multiplication	10 * 5
14	/	Division	10 / 5
14	%	Modulo division	10 % 5
14	**	Exponentiation	10 ** 2
13	+	Addition	10 + 5
13	-	Subtraction	10 - 5
12	<<	Shift left	x << 2
12	>>	Shift right	x >> 2
11	<	Less than	x < y
11	<=	Less than or equal	x <= y
11	>	Greater than	x > y
11	>=	Greater than or equal	x >= y
10	==	Equal	x == y
10	===	Strict equal	x === y
10	!=	Unequal	x != y

10	!==	Strict unequal	x !== y
6	&&	And	x && y
5		Or	x y
3	=	Assignment	x = y
3	+=	Assignment	x += y
3	-=	Assignment	x -= y
3	*=	Assignment	x *= y
3	/=	Assignment	x /= y

24. Ternary operator & conditions

- **Conditions**

1. if

2. else

3. else if

- `someCondition ? thisIfTrue : thisIfFalse`

```
var result= number>40 ? "pass" : "fail" ;
```

```
var test=false;
```

```
console.log("result is: " + test ? "hello" : "hi");
```

25. Switch Statement

Use the switch statement to select one of many blocks of code to be executed.

```
switch(expression) {
```

```
  case n:
```

```
    code block
```

```

    break;
case n:
    code block
    break;
default:
    default code block
}

```

- break & default keyword

26. Loop

- Different Kinds of Loops

for - loops through a block of code a number of times.

for/in - loops through the properties of an object.

while - loops through a block of code while a specified condition is true.

do/while - also loops through a block of code while a condition is true

- For loop

```

for (i = 0; i < 5; i++) {
    text += "The number is " + i + "<br>";
}

```

- For loop

```

var person = {fname:"John", lname:"Doe", age:25};
var text = "";
var x;
for (x in person) {
    text += person[x];
}

```

- While Loop

```
while (condition) {  
  code block to be executed  
}
```

- Do/While Loop

```
do {  
  code block to be executed  
}  
while (condition);
```

27. Function

- Function declarations

```
function square(number) {  
  return number * number;  
}
```

- Function Expression

```
//Anonymous function expression  
var a=function(num){  
  return 3*num;  
}  
alert(a(8))
```

```
//named function expression  
var a = function bar(num) {  
  return 3;  
}
```

```
//self invoking function expression  
(function sayHello() {
```

```
    alert("hello!");  
  })();
```

28. Function

- Functions are object, they can be used like any other value.
- Functions can be stored in variables, objects, and arrays.
- Functions can be passed as arguments to functions, and functions can be returned from functions.
- A function always returns a value.
- If the return value is not specified, then undefined is returned.
- Named function:

The function can use its name to call itself recursively.

The name can also be used by debuggers and development tools to identify the function.

29. Reference's

- <http://www.w3schools.com/js/>
- <http://www.tutorialspoint.com/javascript/>
- <https://developer.mozilla.org/en-US/docs/Web/JavaScript>
- <https://www.codeschool.com/paths/javascript>