
WORK
TRUE
千锋教育
QIAN FENG EDUCATION
做真实的自己，用良心做教育
JS基础
H5教学部

目录

做真实的自己，用良心做教育

千锋教育
QIAN FENG EDUCATION

JavaScript语言介绍

1. JavaScript 的诞生

JavaScript 诞生于 1995 年。由Netscape(网景公司)的程序员Brendan Eich(布兰登)与Sun公司联手开发一门脚本语言，最初名字叫做Mocha，1995年9月改为LiveScript。12月，Netscape公司与Sun公司（Java语言的发明者）达成协议，后者允许将这种语言叫做JavaScript。这样一来，Netscape公司可以借助Java语言的声势。

1996年3月， Netscape公司的浏览器Navigator 2.0浏览器正式内置了JavaScript脚本语言. 此后其他主流浏览器逐渐开始支持JavaScript.

JavaScript语言介绍

2. JavaScript 的版本

JavaScript这种语言的基本语法结构是由ECMAScript来标准化的, 所以我们说的JavaScript版本一般指的是ECMAScript版本.

1997年7月，ECMAScript 1.0发布。

1998年6月，ECMAScript 2.0版发布。

1999年12月，ECMAScript 3.0版发布。

2007年10月，ECMAScript 4.0版草案想要提交ECMA组织, 但由于4.0版的目标过于激进, 改动太大, 并且微软,谷歌等大公司极力反对；一直到2008年7月ECMA开会决定，中止ECMAScript 4.0的开发（即废除了这个版本）

2009年12月，ECMAScript 5.0版正式发布

2011年6月，ECMAscript 5.1版发布

2015年6月，ECMAScript 6正式发布，并且更名为“ECMAScript 2015”。

JavaScript语言介绍

3. JavaScript 的优势

JavaScript是一门脚本语言, 长期处于计算机语言排行榜的前10位, 且在脚本语言排行榜中长期处于领先地位直至今日, 它的发展前景可想而知.

JavaScript一直伴随着互联网一起发展, 互联网的发展也推动和刺激了JavaScript的发展, 目前苹果公司的Safari, 谷歌的Chrome,微软的IE等几乎全部浏览器都支持JavaScript, 基于JavaScript开发的库和框架数不胜数, 例如: jQuery, PhoneGap, Angular, React, Vue等...

JavaScript将在前端和服务器端(Node.js)有更好的发展

JavaScript语言介绍

4. JavaScript 的简介

JavaScript是一种专为与网页交互而设计的脚本语言, 具有较强的逻辑性.

JavaScript 是一种具有面向对象能力的、解释型的程序设计语言。更具体一点，它是基于对象和事件驱动并具有相对安全性的客户端脚本语言。因为他不需要在一个语言环境下运行，而只需要支持它的浏览器即可。它的主要目的是，验证发往服务器端的数据、增加 Web互动、加强用户体验度等.

不同于服务器端脚本语言，例如PHP、ASP(.net)和JSP(Java)，JavaScript主要被作为客户端脚本语言在用户的浏览器上运行，不需要服务器的支持

JavaScript语言介绍

5. JavaScript 语言的特点

(1) 脚本语言。JavaScript是一种解释型的脚本语言,C、C++等语言先编译后执行,而JavaScript是在程序的运行过程中逐行进行解释。

(2) 基于对象。JavaScript是一种基于对象的脚本语言,它不仅可以创建对象,也能使用现有的对象。

(3) 简单。JavaScript语言中采用的是弱类型的变量类型,对使用的数据类型未做出严格的要求.

(4) 动态性。JavaScript是一种采用事件驱动的脚本语言,它不需要经过Web服务器就可以对用户的输入做出响应。在访问一个网页时,鼠标在网页中进行鼠标点击或上下移、窗口移动等操作,JavaScript都可直接对这些事件给出相应的响应。

(5) 跨平台性。JavaScript脚本语言不依赖于操作系统,仅需要浏览器的支持。因此一个JavaScript脚本在编写后可以带到任意机器上使用,前提是机器上的浏览器支持JavaScript脚本语言,目前JavaScript已被大多数的浏览器所支持。

JavaScript语言介绍

JavaScript由三部分组成:

1. 核心(ECMAScript)
2. 浏览器对象模型(BOM)
3. 文档对象模型(DOM)

ECMAScript：是一种由ECMA国际(前身为欧洲计算机制造商协会,英文名称是European Computer Manufacturers Association)通过ECMA-262标准化的脚本程序设计语言。 ECMAScript 定义的只是这门语言的基础，他的组成部分有：语法、类型、语句、关键字、保留字、操作符、对象等

BOM： Browse Object Model, 浏览器对象模型，提供与浏览器交互的方法和接口(API), 开发人员使用BOM可以控制浏览器显示页面以外的部分.

DOM： Document Object Model,文档对象模型，提供访问和操作网页HTML内容的方法和接口

JavaScript语言介绍

JavaScript的编辑工具：写代码的工具

如：**HBuilder**，**Sublime Text**, Dreamweaver，Notepad++, 文本等..

JavaScript的运行环境：看结果的地方

如： chrome, firefox, IE等浏览器

JavaScript语言介绍

1, 导入JavaScript标签: <script type= “text/javascript” ></script>

2, 在标签中间写js代码

第一句javascript代码：alert(“hello world!”) ;

第二句javascript代码：document.write(“亲，我在页面上，跟alert不一样噢！”);

第三句javascript代码：console.log(“我是在控制台打印的, 以后常用我！”);

注意: document.write可以输出任何HTML的代码

3, script标签可以出现多次, 且可以出现在html文件的任何地方, 建议写在<head></head>之间; 另外,同一个文件中Javascript和HTML代码, 它们的执行顺序都是自上而下，谁在前就谁先执行, 谁在后就后执行.

4, JavaScript的注释

单行注释: //, 多行注释 /* */

JavaScript语言介绍

外部javaScript文件引入方式

<script type="text/javascript" src="demo1.js" ></script>

注意： 1、不可以使用单标, 如这是不正确的写法 <script type="text/javascript" src="demo1.js “/ >

2、在引入了外部文件的标签中写代码会无效, 下面的alert()不会执行

<script src= “demo1.js” >alert(‘xxxx’)</script>

<script>标签的属性:

src 表示要引入的外部文件

type 表示脚本语言的类型

language已废弃。原来用于代码使用的脚本语言。由于大多数浏览器忽略它，所以不要用了。

defer：可选。表示脚本可以延迟到文档完全被解析和显示之后再执行。由于大多数浏览器不支持，故很少用。

charset：可选。表示通过 src 属性指定的字符集。由于大多数浏览器忽略它，所以很少有人用它。

JS变量、关键字、命名规范

- 变量定义(使用var关键字)：
  var age;　　//var 是关键字，age是变量名

- 赋值：
  age = 20;

- 定义的同时赋值：
  var age=20 ;

- 可以一次定义多个变量：
  var name= “zhangsan", age=18，weight=108;

- JS是弱数据类型的语言，容错性较高, 在赋值的时候才确定数据类型
  var b ;　　//temp时啥数据类型？不确定
  b = 12;　　//temp变量是数字类型
  b = “hello” ;　　//temp变量变成了字符串类型
  console.log(typeof b);

JS变量、关键字、命名规范

- 关键字：已经被JS内部使用了的

Break	Else	New	var
Case	Finally	Return	void
Catch	For	Switch	while
Continue	Function	This	with
Default	If	Throw	
Delete	In	Try	
Do	Instanceof	Typeof	

- 保留字: 虽然暂时还未被使用, 但将来可能会被JS内部使用

Abstract	Enum	Int	short
Boolean	Export	Interface	static
Byte	Extends	Long	super
Char	Final	Native	synchronized
Class	Float	Package	throws
Const	Goto	Private	transient
Debugger	Implements	Protected	volatile
Double	Import	Public	

JS变量、关键字、命名规范

变量的命名规范

1, 变量名可以是数字,字母,下划线_和美元符$组成;

2, 第一个字符不能为数字

3, 不能使用关键字或保留字

4, 标识符区分大小写，如：age和Age是不同的变量。但强烈不建议用同一个单词的大小写区分两个变量。

5, 变量命名尽量遵守驼峰原则: myStudentScore

6, 变量命名尽量见名思意, 可参考下图

类型	前缀	类型	实例
数组	a	Array	aItems
布尔值	b	Boolean	bIsComplete
浮点数	f	Float	fPrice
函数	fn	Function	fnHandler
整数	i	Integer	iItemCount
对象	o	Object	oDiv1
正则表达式	re	RegExp	reEmailCheck
字符串	s	String	sUserName

JS数据类型

JS数据类型一般可以分为:

- Boolean: 布尔类型
- Number：数字（整数int，浮点数float）
- String：字符串
- Array：数组
- Object：对象

特殊数据类型 Null、Undefined

注意: 变量的类型在赋值时才能确定

JS数据类型

typeof 操作符:

用来检测变量的数据类型, 对于值或变量使用 typeof 操作符会返回如下字符串:

Undefined数据类型的值为：undefined 未定义

Boolean数据类型的值为：boolean 布尔值

String数据类型的值为：string 字符串

Number数据类型的值为：number 数值

Object数据类型的值为：object 对象或者null

Function数据类型的值为：function 函数

JS数据类型

Undefined类型:

Undefined 类型只有一个值，即特殊的 undefined。在使用 var 声明变量，但没有对其初始化时，这个变量的值就是 undefined

例如:

```
var b;  console.log(b);  //undefined
```

注意: 我们在定义变量的时候， 尽可能的不要只声明，不赋值, 而是声明的同时初始化一个值。

JS数据类型

Null 类型:

Null 类型是一个只有一个值的数据类型，即特殊的值 null。它表示一个空对象引用(指针)，而 typeof 操作符检测 null 会返回 object。

例如:

```
var b = null;   console.log(typeof b);
```

undefined 是派生自 null 的，因此 ECMA-262 规定对它们的相等性测试返回 true, 表示值相等， 但是两者的数据类型是不一样的。

例如:

```
console.log(undefined == null);  //true
var b ，car = null;   console.log(typeof b== typeof car); //false
```

JS数据类型

Boolean类型:

Boolean 类型有两个值：true和false。而true一般等于1，false一般等于0。 JavaScript 是区分大小写的，True和False或者其他都不是Boolean类型的值。

例如:

```
var b= true;  console.log(typeof b);
```

Boolean可以将一个值转换为其对应的 Boolean 值，可以使用转型函数Boolean()。

例如:

```
var a = 'Hello World!';
var b = Boolean(a);
console.log(typeof b);
```

JS数据类型

Boolean类型:

Boolean 类型的转换规则: (牢记)

String: 非空字符串为true, 空字符串为false

Number: 非0数值为true, 0或者NaN为false

Object: 对象不为null则为true, null为false

Undefined : undefined为false

JS数据类型

Number类型:

Number 类型包含两种数值：整型和浮点型.

整型:

```
var b = 100;    console.log(b);
```

浮点类型:

就是该数值中必须包含一个小数点，并且小数点后面必须至少有一位数字

```
var b = 3.8;
var b = 0.8;
var b = .8;          //有效，但不推荐此写法
```

JS数据类型

Number类型:

由于保存浮点数值需要的内存空间比整型数值大两倍，因此 ECMAScript 会自动将可以 转换为整型的浮点数值转成为整型。

```
var b = 8.;          //小数点后面没有值，转换为8
var b = 12.0;        //小数点后面是0，转成为12
```

对于那些过大或过小的数值，可以用科学技术法来表示(e 表示法)。用 e 表示该数值的前面 10 的指数次幂。

```
var box = 4.12e9;            //即 4120000000
var box = 0.0000412;         //即 4.12e-5
```

浮点数值的范围在：Number.MIN_VALUE ~ Number.MAX_VALUE 之间, 如果超过了浮点数值范围的最大值或最小值，那么就先出现 Infinity(正无穷)或-Infinity(负无穷)。

JS数据类型

Number类型:

NaN, 即非数值(Not a Number)是一个特殊的值

这个数值用于表示一个本来要返回数值的操作数未返回数值的情况(这样就不会抛出错误了)。比如，在其他语言中，任何数值除 以 0 都会导致错误而终止程序执行。但在ECMAScript中，会返回出特殊的值，因此不会影响程序执行。

```
var b = 0/0;    //NaN
var b = 12/0;  //Infinity
var b = 12/0 * 0;  //NaN
```

ECMAScript 提供了 isNaN()函数，用来判断是不是 NaN。isNaN()函数在接收到一个值之后，会尝试将这个值转换为数值。

```
console.log(isNaN(NaN));        //true
console.log(isNaN(25));         //false，25 是一个数值
console.log(isNaN('25'));       //false，'25'是一个字符串数值，可以转成数值
console.log(isNaN('zhangsan'));     //true，'zhangsan'不能转换为数值
console.log(isNaN(true));        //false       true 可以转成成 1
```

JS数据类型

字符串转换数字类型：

parseInt()　　是把其它类型转换为整型

parseFloat()　是把其它类型转换为浮点型（小数）

Math.round() 四舍五入

如：(78.566) -> 78

JS运算符的使用 - 算术运算符

- 算术运算符 ： + ，-, *, /, %(取余数)
- 字符串和变量的拼接： +
- 关系运算符 ： <、>、<=、>=、==、===、!=, !==
- 逻辑运算符 ： && 与(且)、|| 或、！非
- 赋值运算符 ： =、+=、-=、*=、/=、%=
- 自增、自减 ： ++a, a++, --a, a--

JS运算符的使用 - 算术运算符

算术运算符:

运算符	说 明	示 例	备 注
+	加	a = 5 + 8	
-	减	a = 8 - 5	
/	除	a = 20 / 5	
*	乘	a = 5*19	
%	取模—两个数相除的余数	10 % 3 = 1	
++	一元自加。该运算符带一个操作数，将操作数的值加 1。返回的值取决于 ++ 运算符位于操作数的前面或是后面	++x， x++	++x将返回 x 自加运算后的值。 x++ 将返回 x 自加运算前的值
--	一元自减。该运算符只带一个操作数。返回的值取决于 -- 运算符位于操作数的前面或是后面	--x， x--	--x 将返回 x 自减运算后的值。 x-- 将返回 x 自减运算前的值

JS运算符的使用 - 算术运算符

JS代码规范:

1, 保持代码缩进

2, 变量名遵守命名规范, 尽量见名思意

3, JS语句的末尾尽量写上分号;

4, 运算符两边都留一个空格, 如：var n = 1 + 2;

练习

1, 今天课堂所有的代码，照敲至少一遍, 理解并熟练掌握今日知识点.

2, 入职薪水10K，每年涨幅入职薪水的5%，50年后工资多少？

3, 为抵抗洪水，战士连续作战89小时，编程计算共多少天零多少小时？

4, 小明要到美国旅游，可是那里的温度是以华氏度为单位记录的。它需要一个程序将华氏温度（80度）转换为摄氏度，并以华氏度和摄氏度为单位分别显示该温度。

提示：摄氏度与华氏度的转换公式为：摄氏度 = 5/9.0*(华氏度-32)

5, 给定一个三位数，分别把这个数字的百位、十位、个位算出来并显示。

扩展内容 - 二进制, 八进制, 十进制, 十六进制介绍

进制： 在计算机处理器内部只认识二进制, 在内存存储的数据其实就是大量的开关,每个开关，可以存储一个1或0，称为一位(1bit)，每八位称为1字节(1byte)

1位 : 是二进制的0或者1

1byte = 8位

1KB = 1024byte

1MB = 1024KB

1GB = 1024MB

1TB = 1024GB

..1PB, 1EB

例如: 十进制的23在内存中存储的是二进制10111

23 -> 10111

00000000 00000000 00000000 00010111(内存中存储的23)

扩展内容 - 二进制, 八进制, 十进制, 十六进制介绍

进制:

二进制: 0, 1 (0~1)
八进制: 0, 1, 2, 3, 4, 5, 6, 7 (0~7)
十进制: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 (0~9)
十六进制: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F (0~15)

常用值: 2的次方
2^2 = 4
2^3 = 8
2^4 = 16
2^5 = 32
2^6 = 64
2^7 = 128
2^8 = 256
2^9 = 512
2^10 = 1024
2^11 = 2048

扩展内容 - 二进制, 八进制, 十进制, 十六进制介绍

进制转换:

2进制 :101011

转10进制: (从右往左)1*2⁰ + 1*2¹ + 0*2^2 + 1*2^3 +0*2^4 + 1*2^5 = 1+2+0+8+0+32 = 43

转8进制: (从右往左3个一组 101 011): 53

转16进制: (从右往左4个一组 0010 1011): 2B

10进制: 43

转2进制: 32+8+2+1 = 101011

转16进制: 32+11 = 2B

转8进制: 40+3 = 53

8进制: 53

转10进制: 5*8¹ + 3*8⁰ = 43

16进制: 2B

转10进制: 2*16¹ + 11*16⁰ = 32+11 = 43

8进制和16进制转非10进制,可以先转换成10进制,再由10进制转

扩展内容 - 二进制, 八进制, 十进制, 十六进制介绍

扩展练习

1. 将这些二进制1100 , 10111转成八进制、十进制、十六进制
2. 将下列10进制数转成二进制

   193， 49， 81， 35
3. 将下列二进制数转成十进制数

   100001001

   11001101

THANK YOU

千锋教育
QIAN FENG EDUCATION

做真实的自己，用良心做教育