36 javascript

JAVAEE高级

一:javascript的概述

> 1、JavaScript是web上一种功能强大的编程语言，用于开发交互式的web页面
>
> 2、它不需要进行编译，而是直接嵌入在HTML页面中，由浏览器执行
>
> 3、JavaScript被设计用来向HTML页面添加交互行为。
>
> 4、JavaScript是一种脚本语言(轻量级的编程语言)
>
> 5、JavaScript由数行可执行计算机代码组成。
>
> 6、JavaScript通常被直接嵌入HTML页面。
>
> 7、JavaScript是一种解释性语言(不进行预编译)。

二：作用

> 1、嵌入动态文本于HTML页面
>
> 2、对浏览器事件做出响应、读写HTML元素
>
> 3、验证提交数据、检测访客的浏览器信息等

三：引入的方式

> 1、内嵌式
>
> 在HTML文档直接嵌入JavaScript脚本

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
    </body>
   <script type="text/javascript">
     alert("执行了")
   </script>
</html>
```

2、外链式
链接外部JavaScript脚本文件

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
</body>
<script type="text/javascript" src="js/slider.js"></script>
</html>
```

四：基本变量语法

1、在使用javaScript的时候，遵循以下命名规范

2、必须以字母或下划线开头，中间可以是数字、字符或下划线
，也能以 $ 和 _ 符号开头（不过我们不推荐这么做）

3、变量名不能包含空格等符号，严格区分大小写

4、不能使用javaScript关键字作为变量名，如：function

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		

1、变量的声明

1、变量是用于存储信息的"容器"

2、在 JavaScript 中创建变量通常称为"声明"变量

3、我们使用 var 关键词来声明变量

```
var carname;
```

4、变量声明之后，该变量是空的（它没有值）如需向变量赋值，请使用等号

```
carname="Volvo";
```

5、不过，您也可以在声明变量时对其赋值

```
var carname="Volvo";
```

```
<script>
var x=5;
var y=6;
var z=x+y;
document.write(x + "<br>");
document.write(y + "<br>");
document.write(z + "<br>");
</script>
```

2、多变量

> 1、可以在一条语句中声明很多变量。该语句以 var 开头，并使用逗号分隔变量即可

```
var lastname="Doe", age=30, job="carpenter";
```

3、重新声明 JavaScript 变量

> 1、如果重新声明 JavaScript 变量，该变量的值不会丢失：
>
> 2、在以下两条语句执行后，变量 carname 的值依然是 "Volvo"：

```
var carname="Volvo";
var carname;
```

五：输出

> 1、JavaScript 可以通过不同的方式来输出数据
>
> 2、window.alert() 弹出警告框
>
> 3、 document.write() 方法将内容写到 HTML 文档中
>
> 4、innerHTML 写入到 HTML 元素

> 5、 console.log() 写入到浏览器的控制台

1、 window.alert()

> 1、弹出警告框来显示数据

```
    <script type="text/javascript">
window.alert(5 + 6);
    </script>
```

2、HTML 元素

> 1、JavaScript 访问某个 HTML 元素，可以使用 document.getElementById(id)
>
> 2、使用 "id" 属性来标识 HTML 元素，并 innerHTML 来获取或插入元素内容

```
    <body>
        <p id="demo">我的第一个段落。</p>
    </body>
    <script>
        document.getElementById("demo").innerHTML = "段落已修改。";
    </script>
```

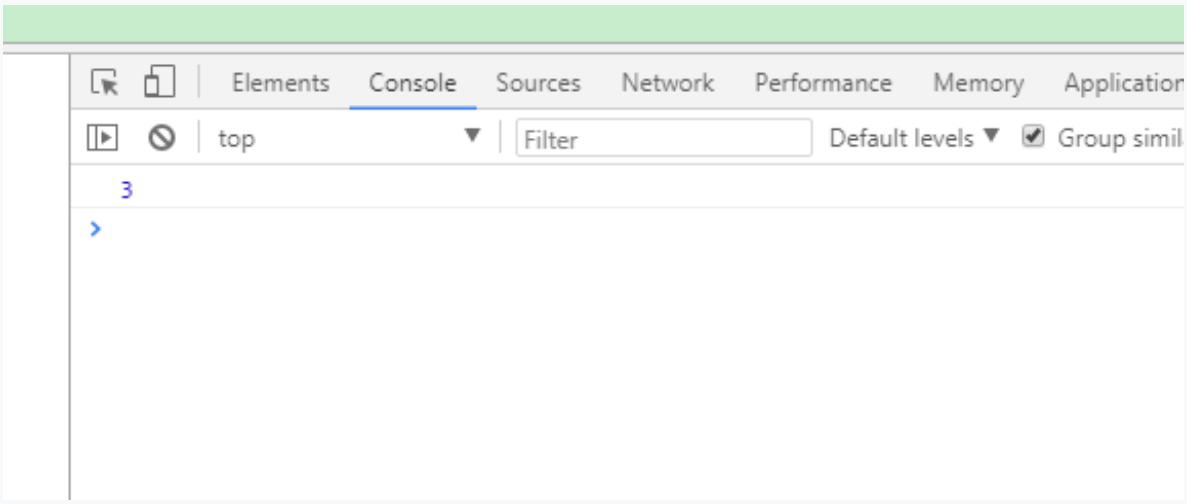
3、写到 HTML 文档

> 1、一般用于测试目的，您可以将JavaScript直接写在HTML 文档中

```
    <script>
        document.write(Date());
    </script>
</html>
```

4、写到控制台

```
    <script>
        a = 1;
        b = 2;
        c = a + b;
        console.log(c);
    </script>
</html>
```



六：标示符

1、javaScript 语句通常以一个 语句标识符 为开始，并执行该语句

2、语句标识符是保留关键字不能作为变量名使用

语句	描述
break	用于跳出循环。
catch	语句块，在 try 语句块执行出错时执行 catch 语句块。
continue	跳过循环中的一个迭代。
do ... while	执行一个语句块，在条件语句为 true 时继续执行该语句块。
for	在条件语句为 true 时，可以将代码块执行指定的次数。
for ... in	用于遍历数组或者对象的属性（对数组或者对象的属性进行循环操作）。
function	定义一个函数
if ... else	用于基于不同的条件来执行不同的动作。
return	退出函数
switch	用于基于不同的条件来执行不同的动作。
throw	抛出（生成）错误。
try	实现错误处理，与 catch 一同使用。
var	声明一个变量。
while	当条件语句为 true 时，执行语句块。

七：JavaScript 注释

> 1、JavaScript 不会执行注释
>
> 2、我们可以添加注释来对 JavaScript 进行解释，或者提高代码的可读性

1、单行注释

> 1、以 // 开头

```
    <script>
        // 输出标题：
        document.getElementById("demo").innerHTML = "欢迎来到我的主页";
    </script>
```

2、 多行注释

> 1、多行注释以 /* 开始，以 */ 结尾。

```
    <script>
        /*
        下面的这些代码会输出
        一个标题和一个段落
        并将代表主页的开始
        */
        document.getElementById("demo").innerHTML = "欢迎来到我的主页";
    </script>
```

八：JavaScript数据类型

> 1、值类型(基本类型)：字符串（ String ）、数字(Number)、布尔(Boolean)、对空（ Null ）、未定义（ Undefined ）
>
> 2、引用数据类型：对象(Object)、数组(Array)、函数(Function)。

1、动态类型

> 1、JavaScript 拥有动态类型
>
> 2、这意味着相同的变量可用作不同的类型

```
    <script>
        var x; // x 为 undefined
        var x = 5; // 现在 x 为数字
        var x = "醉红尘"; // 现在 x 为字符串
        document.getElementById("demo").innerHTML = x;
    </script>
```

2、字符串

> 1、字符串可以是引号中的任意文本可以使用单引号或双引号

```
<script>
    var carname1="Volvo XC60";
    var carname2='Volvo XC60';
    var answer1='It\'s alright';
    var answer2="He is called \"Johnny\"";
    var answer3='He is called "Johnny"';
    document.write(carname1 + "<br>")
    document.write(carname2 + "<br>")
    document.write(answer1 + "<br>")
    document.write(answer2 + "<br>")
    document.write(answer3 + "<br>")
</script>
```

3、数字

> 1、JavaScript 只有一种数字类型数字可以带小数点，也可以不带

```
    <script>
        var x1 = 34.00;
        var x2 = 34;
        var y = 1235;
        var z = 123 - 5;
        document.write(x1 + "<br>")
        document.write(x2 + "<br>")
        document.write(y + "<br>")
        document.write(z + "<br>")
    </script>
```

4、布尔值

> 1、布尔（逻辑）只能有两个值：true 或 false

```
    <script>
        var x = true;
        var y = false;
```

```
        document.write(y+ "<br>")
        document.write(x + "<br>")
    </script>
```

5、数组

> 1、数组下标是基于零的，所以第一个下标是 `[0]`，第二个是 `[1]`，以此类推

```
<script>
        //方式一
        var cars1 = new Array();
        cars1[0] = 1;
        cars1[1] = 3;
        cars1[2] = 2;
        //方式二
        var cars2 = new Array("Saab", "Volvo", "BMW");
        //方式三
        var cars3 = ["Saab", "Volvo", "BMW"];
        for(i = 0; i < cars1.length; i++) {
            document.write(cars1[i] + "<br>");
        }
    </script>
```

6、Undefined 和 Null

> 1、Undefined 这个值表示变量不含有值
>
> 2、可以通过将变量的值设置为 null 来清空变量

```
    <script>
        var person;
        var car = "java";
        document.write(person + "<br>");
        document.write(car + "<br>");
        var car = null
        document.write(car + "<br>");
    </script>
```

7、声明变量类型

> 1、当您声明新变量时，可以使用关键词 "new" 来声明其类型

```
var carname=new String;
var x=      new Number;
var y=      new Boolean;
var cars=   new Array;
var person= new Object;
```

九：函数

> 1、函数是由事件驱动的或者当它被调用时执行的可重复使用的代码块
>
> 2、函数就是包裹在花括号中的代码块，前面使用了关键词 function
>
> 3、JavaScript 对大小写敏感
>
> 4、关键词 function必须是小写的，并且必须以与函数名称相同的大小写来调用函数

```
    <body>
        <button onclick="myFunction()">点我</button>
    </body>
    <script>
        function myFunction() {
            alert("Hello World!");
        }
    </script>
```

1、带参数的函数

> 1、在调用函数时，您可以向其传递值，这些值被称为参数
>
> 2、可以发送任意多的参数，由逗号 (,) 分隔
>
> 3、当您声明函数时，把参数作为变量来声明

```
<script>
    var var1=1,var2=2;
    function myFunction(var1,var2) {
        alert(var1);
        alert(var2);
    }
    myFunction(var1,var2);
</script>
```

> 4、变量和参数必须以一致的顺序出现。第一个变量就是第一个被传递的参数的给定的值，以此类推

```
<body>
    <button onclick="myFunction('qqq','sSc')">点我</button>
</body>
<script>
    function myFunction(var1,var2) {
        alert(var1);
        alert(var2);
    }
</script>
```

2、返回值的函数

> 1、希望函数将值返回调用它的地方，通过使用 return 语句就可以实现
>
> 2、在使用 return 语句时，函数会停止执行，并返回指定的值

```
<script>
    function myFunction() {
        var x = 5;
        return x;
    }
var v=  myFunction();
document.write(v);
</script>
```

> 3、整个 JavaScript 并不会停止执行，仅仅是函数。JavaScript 将继续执行代码，从调

> 用函数的地方

3、函数调用

> 1、函数中的代码在函数被调用后执行

方式一：作为一个函数调用

> 1、以下函数不属于任何对象
>
> 2、但是在 JavaScript 中它始终是默认的全局对象
>
> 3、全局对象是 HTML 页面本身，所以函数是属于 HTML 页面
>
> 4、页面对象是浏览器窗口(window 对象)

```
    <script>
        function myFunction() {
            alert(111);
        }
        myFunction();
    </script>
```

> 5、以上函数会自动变为 window 对象的函数
>
> 6、myFunction() 和 window.myFunction() 是一样

```
    <script>
        function myFunction() {
            alert(111);
        }
        window.myFunction();
    </script>
```

方式二：函数作为方法调用

> 1、 JavaScript 中可以将函数定义为对象的方法

```
    <script>
        var myObject = {
            firstName: "John",
            lastName: "Doe",
            fullName: function() {
                return this.firstName + " " + this.lastName;
            }
        }
        alert(myObject.fullName());  // 返回 "John Doe"
    </script>
```

方式三：使用构造函数调用函数

> 1、如果函数调用前使用了 new 关键字, 则是调用了构造函数
>
> 2、看起来就像创建了新的函数，但实际上 JavaScript 函数是重新创建的对象

```
    <script>
        // 构造函数:
        function myFunction(arg1, arg2) {
            this.firstName = arg1;
            this.lastName = arg2;
        }
        var x = new myFunction("John", "Doe");
        alert(x.firstName); // 返回 "John Doe"
    </script>
```

4、局部 JavaScript 变量

> 1、在 JavaScript 函数内部声明的变量（使用 var）是局部变量，所以只能在函数内部访问它。（该变量的作用域是局部的）。
>
> 3、您可以在不同的函数中使用名称相同的局部变量，因为只有声明过该变量的函数才能识别出该变量。
>
> 4、只要函数运行完毕，本地变量就会被删除。

5、全局 JavaScript 变量

> 1、在函数外声明的变量是全局变量，网页上的所有脚本和函数都能访问它

6、JavaScript 变量的生存期

> 1、JavaScript 变量的生命期从它们被声明的时间开始
>
> 2、局部变量会在函数运行以后被删除
>
> 3、全局变量会在页面关闭后被删除

7、向未声明的 JavaScript 变量分配值

> 1、如果您把值赋给尚未声明的变量，该变量将被自动作为 window 的一个属性。
> 这条语句：

```
carname="Volvo";
//将声明 window 的一个属性 carname。
```

> 2、非严格模式下给未声明变量赋值创建的全局变量，是全局对象的可配置属性，可以删除

```
    <script>
        var var1 = 1; // 不可配置全局属性
        var2 = 2; // 没有使用 var 声明，可配置全局属性

        document.write(window.var1);
        document.write(this.var1);

        delete var1; // false 无法删除
        document.write(var1);

        delete var2;
        document.write(delete var2); // true
    </script>
```