# JavaScript学习

## 1、为什么学习JavaScript

### 一、学习的价值

1. 所有主流浏览器都支持JavaScript。

2. 目前，全世界大部分网页都使用JavaScript。

3. 它可以让网页呈现各种动态效果。

4. 做为一个Web开发师，如果你想提供漂亮的网页、令用户满意的上网体验，JavaScript是必不可少的工具。

### 二、易学行

1.学习环境无外不在，只要有文本编辑器，就能编写JavaScript程序。

2.我们可以用简单命令，完成一些基本操作。

### 三、从哪开始学习呢？

学习JavaScript的起点就是处理网页，所以我们先学习基础语法和如何使用DOM进行简单操作。

## 2、JavaScript应该放在哪里

### 一、放入html文件中

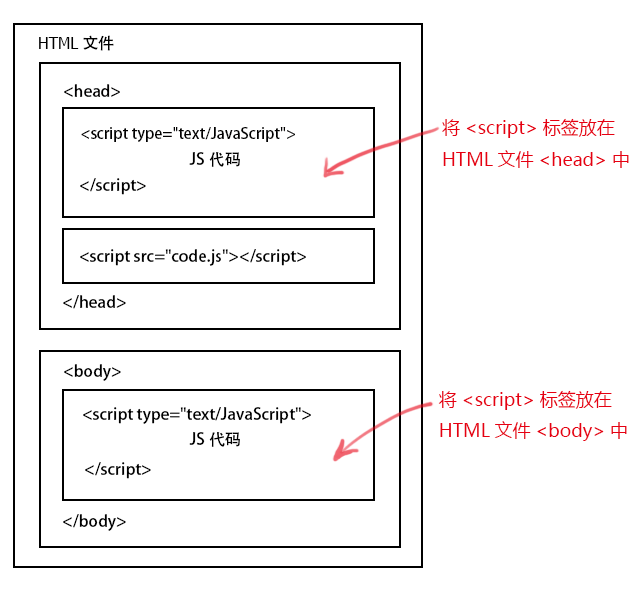
我们可以将JavaScript代码放在html文件中任何位置，但是我们一般放在网页的head或者body部分。

**放在<head>部分**

最常用的方式是在页面中head部分放置<script>元素，浏览器解析head部分就会执行这个代码，然后才解析页面的其余部分。

**放在<body>部分**

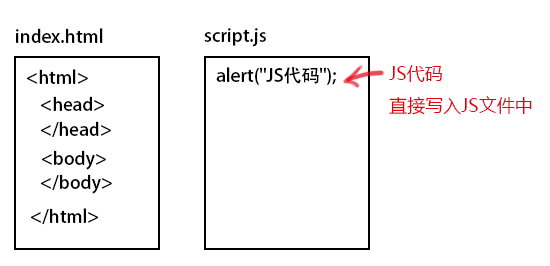
JavaScript代码在网页读取到该语句的时候就会执行。



注意: javascript作为一种脚本语言可以放在html页面中任何位置，但是浏览器解释html时是按先后顺序的，所以前面的script就先被执行。比如进行页面显示初始化的js必须放在head里面，因为初始化都要求提前进行（如给页面body设置css等）；而如果是通过事件调用执行的function那么对位置没什么要求的。

### 二、引用外部文件

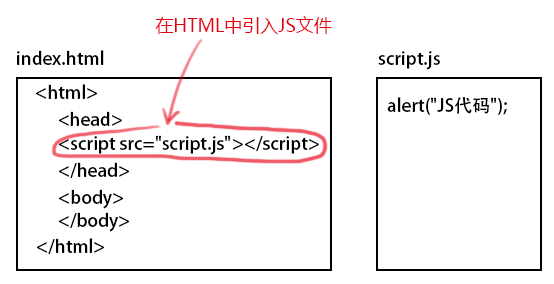
JavaScript代码只能写在HTML文件中吗?当然不是，我们可以把HTML文件和JS代码分开,并单独创建一个JavaScript文件(简称JS文件),其文件后缀通常为.js，然后将JS代码直接写在JS文件中。



注意:在JS文件中，不需要<script>标签,直接编写JavaScript代码就可以了。

JS文件不能直接运行，需嵌入到HTML文件中执行，我们需在HTML中添加如下代码，就可将JS文件嵌入HTML文件中。

<script src="script.js"></script>



## 3、JavaScript基础中的基础

### 一、JavaScript-认识语句和符号

JavaScript语句是发给浏览器的命令。这些命令的作用是告诉浏览器要做的事情。

每一句JavaScript代码格式: 语句;

先来看看下面代码

<script type="text/javascript">

alert("hello!");

</script>

例子中的alert("hello!");就是一个JavaScript语句。

一行的结束就被认定为语句的结束，通常在结尾加上一个分号";"来表示语句的结束。

看看下面这段代码,有三条语句，每句结束后都有";"，按顺序执行语句。

<script type="text/javascript">

document.write("I");

document.write("love");

document.write("JavaScript");

</script>

注意:

1. “;”分号要在英文状态下输入，同样，JS中的代码和符号都要在英文状态下输入。

2. 虽然分号“;”也可以不写，但我们要养成编程的好习惯，记得在语句末尾写上分号。

### 二、注释

注释的作用是提高代码的可读性，帮助自己和别人阅读和理解你所编写的JavaScript代码，注释的内容不会在网页中显示。注释可分为单行注释与多行注释两种。

我们为了方便阅读，注释内容一般放到需要解释语句的结尾处或周围。

单行注释，在注释内容前加符号 “//”。

<script type="text/javascript">

document.write("单行注释使用'//'"); // 我是注释，该语句功能在网页中输出内容

</script>

多行注释以"/\*"开始，以"\*/"结束。

<script type="text/javascript">

document.write("多行注释使用/\*注释内容\*/");

/\*

多行注释

养成书写注释的良好习惯

\*/

</script>

### 三、变量和操作符

**1、变量**

什么是变量? 从字面上看，变量是可变的量；从编程角度讲，变量是用于存储某种/某些数值的存储器。我们可以把变量看做一个盒子，为了区分盒子，可以用BOX1,BOX2等名称代表不同盒子，BOX1就是盒子的名字（也就是变量的名字）。

定义变量使用关键字var,语法如下：

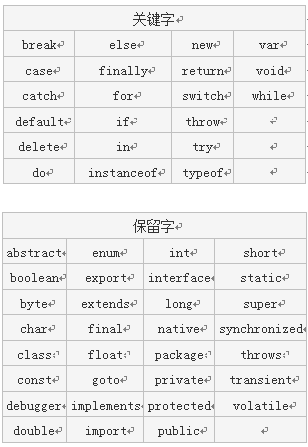
var 变量名

**变量名可以任意取名，但要遵循命名规则:**

1.变量必须使用字母、下划线(\_)或者美元符($)开始。

2.然后可以使用任意多个英文字母、数字、下划线(\_)或者美元符($)组成。

3.不能使用JavaScript关键词与JavaScript保留字。



变量要先声明再赋值，如下：

var mychar;

mychar="javascript";

var mynum = 6;

变量可以重复赋值，如下：

var mychar;

mychar="javascript";

mychar="hello";

**2、操作符**

操作符是用于在JavaScript中指定一定动作的符号。

（1）操作符

看下面这段JavaScript代码。

sum = numa + numb;

其中的"="和"+"都是操作符。

JavaScript中还有很多这样的操作符，例如，算术操作符(+、-、\*、/等)，比较操作符(<、>、>=、<=等)，逻辑操作符(&&、||、！)。

注意: “=” 操作符是赋值，不是等于。

(2) "+"操作符

算术运算符主要用来完成类似加减乘除的工作，在JavaScript中，“+”不只代表加法，还可以连接两个字符串，例如：

mystring = "Java" + "Script"; // mystring的值“JavaScript”这个字符串

（3）、自加一，自减一 ( ++和- -)

算术操作符除了(+、-、\*、/)外，还有两个非常常用的操作符，自加一“++”；自减一“--”。首先来看一个例子：

mynum = 10;

mynum++; //mynum的值变为11

mynum--; //mynum的值又变回10

上面的例子中，mynum++使mynum值在原基础上增加1，mynum--使mynum在原基础上减去1,其实也可以写成:

mynum = mynum + 1;//等同于mynum++

mynum = mynum - 1;//等同于mynum--

（4）、比较操作符

例子：

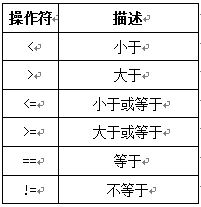
我们先来做道数学题，数学考试成绩中，小明考了90分，小红考了95分，问谁考的分数高?

答: 因为“95 > 90”，所以小红考试成绩高。

其中大于号">" 就是比较操作符，小红考试成绩和小明考试成绩就是操作数，并且是两个操作数。

也就是说两个操作数通过比较操作符进行比较，得到值为真（true）和假(false)。

在JavaScript中，这样的比较操作符有很多，这些操作符的含义如下:



看看下面例子:

var a = 5;//定义a变量，赋值为5

var b = 9; //定义b变量，赋值为9

document.write (a<b); //a小于b的值吗? 结果是真(true)

document.write (a>=b); //a大于或等于b的值吗? 结果是假(false)

document.write (a!=b); //a不等于b的值吗? 结果是真(true)

document.write (a==b); //a等于b的值吗? 结果是假(false)

（5）、逻辑与操作符

**1）、与 &&**

数学里面的“a>b”，在JavaScript中还表示为a>b；数学中的“b大于a，b小于c”是“a<b<c”，那么在JavaScript中可以用&&表示，如下：

b>a && b<c //“&&”是并且的意思, 读法"b大于a"并且" b小于c "

**2）、|| 或**

"||"逻辑或操作符，相当于生活中的“或者”，当两个条件中有任一个条件满足，“逻辑或”的运算结果就为“真”。

例如：本周我们计划出游,可是周一至周五工作,所以周六或者周日哪天去都可以。即两天中只要有一天有空，就可以出游了。

var a=3;

var b=5;

var c;

c=b>a ||a>b; //b>a是true，a>b是false，c是true

**3）、！ 非**

"!"是逻辑非操作符，也就是"不是"的意思,非真即假，非假即真。好比小华今天买了一个杯子，小明说:"杯子是白色的"，小亮说:“杯子是红色的”，小华说："小明说的不是真话，小亮说的不是假话"。猜猜小华买的什么颜色的杯子，答案：红色杯子。

例如：

var a=3;

var b=5;

var c;

c=!(b>a); // b>a值是true,! (b>a）值是false

c=!(b<a); // b<a值是false, ! (b<a）值是true

（6）、操作符优先级

操作符之间的优先级（高到低）:

算术操作符 → 比较操作符 → 逻辑操作符 → "="赋值符号

如果同级的运算是按从左到右次序进行,多层括号由里向外。

var numa=3;

var numb=6;

jq= numa + 30 >10 && numb \* 3<2; //结果为false

**3、数组**

数组是一个值的集合，每个值都有一个索引号，从0开始，每个索引都有一个相应的值，根据需要添加更多数值。



（1）、创建数组

使用数组之前首先要创建，而且需要把数组本身赋至一个变量。

**创建数组语法：**

var myarray=new Array();



创建数组的同时，还可以为数组指定长度，长度可任意指定。

var myarray= new Array(8); //创建数组，存储8个数据。

注意：

1.创建的新数组是空数组，没有值，如输出，则显示undefined。

2.虽然创建数组时，指定了长度，但实际上数组都是变长的，也就是说**即使指定了长度为8，仍然可以将元素存储在规定长度以外。**

（2）、数组赋值

步骤：

第一步：创建数组var myarr=new Array();

第二步：给数组赋值

myarr[1]=" 张三";

myarr[2]=" 李四";

例如：

下面创建一个数组，用于存储5个人的数学成绩。

var myarray=new Array(); //创建一个新的空数组

myarray[0]=66; //存储第1个人的成绩

myarray[1]=80; //存储第2个人的成绩

myarray[2]=90; //存储第3个人的成绩

myarray[3]=77; //存储第4个人的成绩

myarray[4]=59; //存储第5个人的成绩

注意：数组每个值有一个索引号，从0开始。

**我们还可以用简单的方法创建上面的数组和赋值：**

第一种方法：

var myarray = new Array(66,80,90,77,59);//创建数组同时赋值

第二种方法：

var myarray = [66,80,90,77,59];//直接输入一个数组（称 “字面量数组”）

注意：数组存储的数据可以是任何类型（数字、字符、布尔值等）

**如何向数组增加一个新的元素呢？**

只需使用下一个未用的索引，任何时刻可以不断向数组增加新元素。

myarray[5]=88; //使用一个新索引，为数组增加一个新元素

（3）、数组属性length

如果我们想知道数组的大小，只需引用数组的一个属性length。Length属性表示数组的长度，即数组中元素的个数。

语法：

myarray.length; //获得数组myarray的长度

注意：因为数组的索引总是由0开始，所以一个数组的上下限分别是：0和length-1。如数组的长度是5，数组的上下限分别是0和4。

var arr=[55,32,5,90,60,98,76,54];//包含8个数值的数组arr

document.write(arr.length); //显示数组长度8

document.write(arr[7]); //显示第8个元素的值54

同时，JavaScript数组的length属性是可变的，这一点需要特别注意。

arr.length=10; //增大数组的长度

document.write(arr.length); //数组长度已经变为10

数组随元素的增加，长度也会改变，如下:

var arr=[98,76,54,56,76]; // 包含5个数值的数组

document.write(arr.length); //显示数组的长度5

arr[15]=34; //增加元素，使用索引为15,赋值为34

alert(arr.length); //显示数组的长度16 （也就中间的空值也算了）

（4）、二维数组

一维数组，我们看成一组盒子，每个盒子只能放一个内容。

一维数组的表示: myarray[ ]

二维数组，我们看成一组盒子，不过每个盒子里还可以放多个盒子。

二维数组的表示: myarray[ ][ ]

注意: 二维数组的两个维度的索引值也是从0开始，两个维度的最后一个索引值为长度-1。

1. 二维数组的定义方法一

var myarr=new Array(); //先声明一维

for(var i=0;i<2;i++){ //一维长度为2

myarr[i]=new Array(); //再声明二维

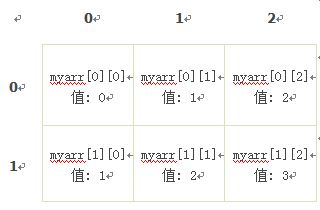
for(var j=0;j<3;j++){ //二维长度为3

myarr[i][j]=i+j; // 赋值，每个数组元素的值为i+j

}

}

将上面二维数组，用表格的方式表示:



2. 二维数组的定义方法二

var Myarr = [[0 , 1 , 2 ],[1 , 2 , 3]]

3. 赋值

myarr[0][1]=5; //将5的值传入到数组中，覆盖原有值。

说明: myarr[0][1] ,0 表示表的行，1表示表的列。

### 四、流程控制语句

**1、if..else语句**

**注意：if小写，大写字母（IF）会出错！**

if...else语句是在指定的条件成立时执行代码，在条件不成立时执行else后的代码。

语法:

if(条件)

{ 条件成立时执行的代码 }

else

{ 条件不成立时执行的代码 }

假设我们通过年龄来判断是否为成年人，如年龄大于等于18岁，是成年人，否则不是成年人。代码表示如下:

<script type="text/javascript">

var myage = 18;

if(myage>=18) //myage>=18是判断条件

{ document.write("你是成年人。");}

else //否则年龄小于18

{ document.write("未满18岁，你不是成年人。");}

</script>

多重判断：

语法:

if(条件1)

{ 条件1成立时执行的代码}

else if(条件2)

{ 条件2成立时执行的代码}

...

else if(条件n)

{ 条件n成立时执行的代码}

else

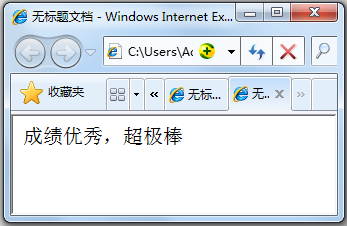
{ 条件1、2至n不成立时执行的代码}

例子：假设数学考试，小明考了86分，给他做个评价，60分以下的不及格，60(包含60分)-75分为良好，75(包含75分)-85分为很好，85(包含85分)-100优秀。

代码表示如下:



结果:



**2、Switch语句**

### 五、函数

是完成某个特定功能的一组语句。如没有函数，完成任务可能需要五行、十行、甚至更多的代码。这时我们就可以把完成特定功能的代码块放到一个函数里，直接调用这个函数，就省重复输入大量代码的麻烦。

如何定义一个函数呢？基本语法如下:

function 函数名()

{

函数代码;

}

说明:

1. function定义函数的关键字。

2. "函数名"你为函数取的名字。

3. "函数代码"替换为完成特定功能的代码。

我们来编写一个实现两数相加的简单函数,并给函数起个有意义的名字：“add2”，代码如下：

function add2(){

var sum = 3 + 2;

alert(sum);

}

函数调用:

函数定义好后，是不能自动执行的，所以需调用它,只需直接在需要的位置写函数就ok了,代码如下:



### 六、打印输出

document.write() 可用于直接向 HTML 输出流写内容。简单的说就是直接在网页中输出内容。

第一种:输出内容用""括起，直接输出""号内的内容。

<script type="text/javascript">

document.write("I love JavaScript！"); //内容用""括起来，""里的内容直接输出。

</script>

第二种:通过变量，输出内容

<script type="text/javascript">

var mystr="hello world!";

document.write(mystr); //直接写变量名，输出变量存储的内容。

</script>

第三种:输出多项内容，内容之间用+号连接。

<script type="text/javascript">

var mystr="hello";

document.write(mystr+"I love JavaScript"); //多项内容之间用+号连接

第四种:输出HTML标签，并起作用，标签使用""括起来。

<script type="text/javascript">

var mystr="hello";

document.write(mystr+"<br>");//输出hello后，输出一个换行符

document.write("JavaScript");

</script>

第五种 输出空格

在写JS代码的时候，大家可以会发现这样现象:

document.write(" 1 2 3 ");

结果: 1 2 3

无论在输出的内容中什么位置有多少个空格，显示的结果好像只有一个空格。

这是因为浏览器显示机制，对手动敲入的空格，将连续多个空格显示成1个空格。

解决方法:

1. 使用输出html标签&nbsp;来解决

document.write("&nbsp;&nbsp;"+"1"+"&nbsp;&nbsp;&nbsp;&nbsp;"+"23");

结果: 1 23

2. 使用CSS样式来解决

document.write("<span style='white-space:pre;'>"+" 1 2 3 "+"</span>");

结果: 1 2 3

在输出时添加“white-space:pre;”样式属性。这个样式表示"空白会被浏览器保留"

### 七、警告（alert 消息对话框）

我们在访问网站的时候，有时会突然弹出一个小窗口，上面写着一段提示信息文字。如果你不点击“确定”，就不能对网页做任何操作，这个小窗口就是使用alert实现的。

语法:

alert(字符串或变量);

看下面的代码:

<script type="text/javascript">

var mynum = 30;

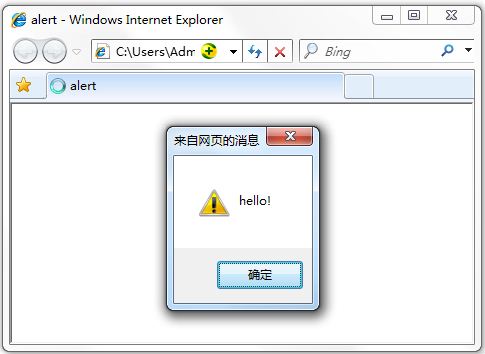
alert("hello!");

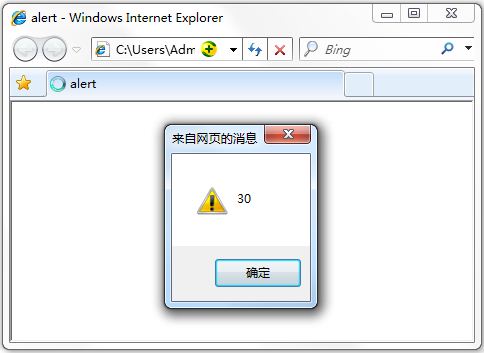
alert(mynum);

</script>

注:alert弹出消息对话框(包含一个确定按钮)。

结果:按顺序弹出消息框





注意:

1. 在点击对话框"确定"按钮前，不能进行任何其它操作。

2. 消息对话框通常可以用于调试程序。

3. alert输出内容，可以是字符串或变量，与document.write 相似。

### 八、确认（confirm 消息对话框）

confirm 消息对话框通常用于允许用户做选择的动作，如：“你对吗？”等。弹出对话框(包括一个确定按钮和一个取消按钮)。

语法:

confirm(str);

参数说明:

str：在消息对话框中要显示的文本

返回值: Boolean值

返回值:

当用户点击"确定"按钮时，返回true

当用户点击"取消"按钮时，返回false

注: 通过返回值可以判断用户点击了什么按钮

看下面的代码:

<script type="text/javascript">

var mymessage=confirm("你喜欢JavaScript吗?");

if(mymessage==true)

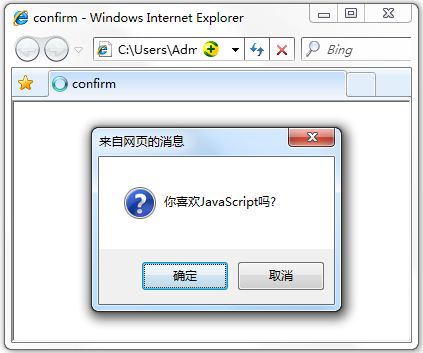
{ document.write("很好,加油!"); }

else

{ document.write("JS功能强大，要学习噢!"); }

</script>

结果:



**注: 消息对话框是排它的，即用户在点击对话框按钮前，不能进行任何其它操作。**

### 九、提问（prompt 消息对话框）

prompt弹出消息对话框,通常用于询问一些需要与用户交互的信息。弹出消息对话框（包含一个确定按钮、取消按钮与一个文本输入框）。

语法:

prompt(str1, str2);

参数说明：

str1: 要显示在消息对话框中的文本，不可修改

str2：文本框中的内容，可以修改

返回值:

1. 点击确定按钮，文本框中的内容将作为函数返回值

2. 点击取消按钮，将返回null

看看下面代码:

var myname=prompt("请输入你的姓名:");

if(myname!=null)

{ alert("你好"+myname); }

else

{ alert("你好 my friend."); }

结果:

注:在用户点击对话框的按钮前，不能进行任何其它操作。

### 十、打开和关闭窗口（window.open）

**打开窗口：**

open() 方法可以查找一个已经存在或者新建的浏览器窗口。

语法：

window.open([URL], [窗口名称], [参数字符串])

参数说明:

URL：可选参数，在窗口中要显示网页的网址或路径。如果省略这个参数，或者它的值是空字符串，那么窗口就不显示任何文档。

窗口名称：可选参数，被打开窗口的名称。

1.该名称由字母、数字和下划线字符组成。

2."\_top"、"\_blank"、"\_self"具有特殊意义的名称。

\_blank：在新窗口显示目标网页

\_self：在当前窗口显示目标网页

\_top：框架网页中在上部窗口中显示目标网页

3.相同 name 的窗口只能创建一个，要想创建多个窗口则 name 不能相同。

4.name 不能包含有空格。

参数字符串：可选参数，设置窗口参数，各参数用逗号隔开。

参数表:



例如:打开http://www.imooc.com网站，大小为300px \* 200px，无菜单，无工具栏，无状态栏，有滚动条窗口：

<script type="text/javascript"> window.open('http://www.imooc.com','\_blank','width=300,height=200,menubar=no,toolbar=no, status=no,scrollbars=yes')

</script>

注意：运行结果考虑浏览器兼容问题。

**关闭窗口：**

close()关闭窗口

用法：

window.close(); //关闭本窗口

或

<窗口对象>.close(); //关闭指定的窗口

例如:关闭新建的窗口。

<script type="text/javascript">

var mywin=window.open('http://www.imooc.com'); //将新打的窗口对象，存储在变量mywin中

mywin.close();

</script>

注意:上面代码在打开新窗口的同时，关闭该窗口，看不到被打开的窗口。

### 十一、DOM操作

**1、认识DOM**

文档对象模型DOM（Document Object Model）定义访问和处理HTML文档的标准方法。DOM 将HTML文档呈现为带有元素、属性和文本的树结构（节点树）。

先来看看下面代码:



将HTML代码分解为DOM节点层次图:



HTML文档可以说由节点构成的集合，三种常见的DOM节点:

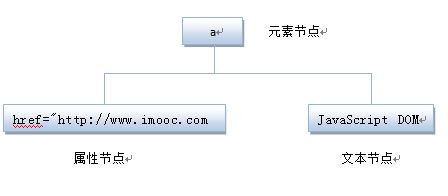
1. 元素节点：上图中<html>、<body>、<p>等都是元素节点，即标签。

2. 文本节点:向用户展示的内容，如<li>...</li>中的JavaScript、DOM、CSS等文本。

3. 属性节点:元素属性，如<a>标签的链接属性href="http://www.imooc.com"。

看下面代码:

<a href="http://www.imooc.com">JavaScript DOM</a>



**2、通过ID获取元素**

学过HTML/CSS样式，都知道，网页由标签将信息组织起来，而标签的id属性值是唯一的，就像是每人有一个身份证号一样，只要通过身份证号就可以找到相对应的人。那么在网页中，我们通过id先找到标签，然后进行操作。

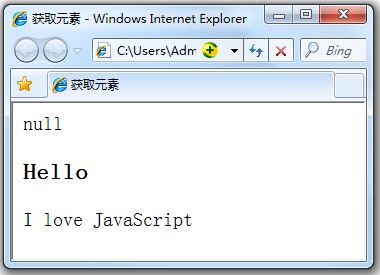
语法:

document.getElementById(“id”)

看看下面代码:



结果:null或[object HTMLParagraphElement]



注:获取的元素是一个对象，如想对元素进行操作，我们要通过它的属性或方法。

**3、innerHTML 属性**

innerHTML 属性用于获取或替换 HTML 元素的内容。

语法:

Object.innerHTML

注意:

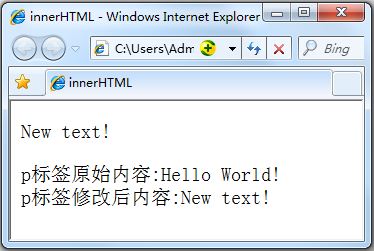
1.Object是获取的元素对象，如通过document.getElementById("ID")获取的元素。

2.注意书写，innerHTML区分大小写。

我们通过id="con"获取<p> 元素，并将元素的内容输出和改变元素内容，代码如下:



结果:



**4、改变 HTML 样式**

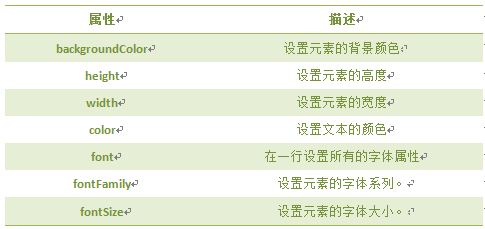
HTML DOM 允许 JavaScript 改变 HTML 元素的样式。如何改变 HTML 元素的样式呢？

语法:

Object.style.property=new style;

注意:Object是获取的元素对象，如通过document.getElementById("id")获取的元素。

基本属性表（property）:



注意:该表只是一小部分CSS样式属性，其它样式也可以通过该方法设置和修改。

看看下面的代码:

改变 <p> 元素的样式，将颜色改为红色，字号改为20,背景颜色改为蓝：

<p id="pcon">Hello World!</p>

<script>

var mychar = document.getElementById("pcon");

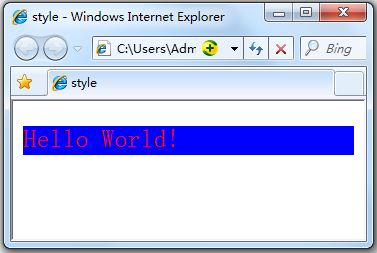
mychar.style.color="red";

mychar.style.fontSize="20";

mychar.style.backgroundColor ="blue";

</script>

结果:



**5、显示和隐藏（display属性）**

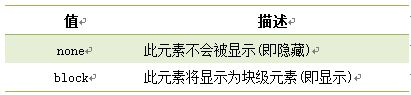
网页中经常会看到显示和隐藏的效果，可通过display属性来设置。

语法：

Object.style.display = value

注意:Object是获取的元素对象，如通过document.getElementById("id")获取的元素。

value取值:



看看下面代码:



**6、控制类名（className 属性）**

className 属性设置或返回元素的class 属性。

语法：

object.className = classname

作用:

1.获取元素的class 属性

2. 为网页内的某个元素指定一个css样式来更改该元素的外观

看看下面代码，获得 <p> 元素的 class 属性和改变className：



结果:

