# Javascript精讲

javascript是属于网络的脚本语言，通常用来操作、控制网页的显示。Javascript是一种轻量级的编程语言，语法和java类似，是前端工程师的必备技能之一。

编写javascript的工具：记事本类工具、eclipse、visual studio、pycharm等。

## javascript基础语法

### 如何在网页中嵌入javascript脚本

**（1）、在页面内部**

**用script标签包起来。**

<script>

window.onload=function(){  
   alert("My First JavaScript");

 }

</script>

通常是放到网页的<head>标签里，或放到页面的最后面。也可以放到页面的中间任何部分，但不推荐。

window.onload加载事件在页面内容加载完成之后立即执行相应的函数。

alert弹出信息提示，常用于调试脚本。

（2）、将脚本放到外部文件

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

### 定于javascript变量

用var来定义变量。

var x=2;

var y=3;

也可以不用var，但不推荐。

Javascript每条语句用分号“；”结束。也可以不用分号，但不推荐。

### Javascript数据类型

Javascript常用类型：整数、浮点数、字符串、布尔类型、对象（内置对象和自定义对象）、null、undefined。

var x=2;

var valueKey=3.56;

var myName=**'**孙悟空**'**;  
var isTearcher = true; // false  
var dt = new Date();  
var key = null;  
var info;

undefined指值未定义，info变量就是这种情况。

实例：dataType.html。

### Javascript注释

（1）、// 双斜杠

// 定义一些变量

var x=2;

var valueKey=3.56;

var myName=**'**孙悟空**'**;

var dt = new Date(); // 日期对象

（2）、/\* \*/ 注释块，常用于多行注释，也可以用于单行注释

/\* 定义一些变量 \*/

var x=2;

var valueKey=3.56;

var myName=**'**孙悟空**'**;  
   
/\* 下面主体脚本代码  
 实现功能：。。。。。。。  
\*/  
var dt = new Date();  
…

### javascript数组

var mycars=new Array();

mycars[0]="Saab"; // 数组元素可以通过下标访问，这里是设置

mycars[1]="Volvo";

mycars[2]="BMW";  
  
var cars=new Array("Saab","Volvo","BMW");  
alert(cars[1]); // 读取第2个元素  
cars=["BMW","Volvo","Saab","Ford"];

### javascript日期对象

（1）、创建日期

var today = new Date();  
var d1 = new Date("October 13, 1975 11:13:00");  
var d2 = new Date(79,5,24);  
var d3 = new Date(79,5,24,11,33,0);

**var** dateStr = today.getFullYear() + **'-'** + (today.getMonth() + 1) + **'-'** + today.getDate()  
 + **' '** + today.getHours() + **':'** + today.getMinutes() + **':'** + today.getSeconds();  
alert(dateStr);

var myDate=new Date();

myDate.setFullYear(2010,0,14);  
myDate.setDate(myDate.getDate()+5);

### Javascript函数

用function来定义：

<!DOCTYPE html>

<html>

<head>

<script>

function myFunction(){

alert("Hello World!");

}

</script>

</head>

<body>

<button onclick="myFunction()">点击这里</button>

</body>

</html>

Html标签的事件属性可以和javascript脚本联系起来。

### 操作网页标签对象

（1）、获取网页标签对象

最常用的就是利用document对象的**getElementById**方法根据网页标签的ID来获取。

<!DOCTYPE html>

<html>

<body>

<h1>我的第一张网页</h1>

<p id="demo">我的第一个段落</p>

<script>

document.getElementById("demo").innerHTML="我的第一段 JavaScript";

</script>

</body>

</html>

其他获取标签对象的函数，如**getElementsByTagName**

getElementsByTagName.html

**function** *getElements*()  
{  
 **var** arrInput = ***document***.getElementsByTagName(**"input"**);  
 *alert*(arrInput.length);  
 **var** obj = arrInput[1];  
 alert(obj.**value**);   
}

<input name="myInput" type="text" size="20" value="第一个文本框" /><br />

<input name="myInput" type="text" size="20" value="第二个文本框" /><br />

<input name="myInput" type="text" size="20" value="第三个文本框" /><br />

<br />

<input type="button" onclick="getElements()"

value="How many input elements?" />

（2）、修改网页标签的样式属性

实例：changeCssStyle.html

（3）、修改网页标签的样式名

实例：changeClassName.html

（4）、批量复选框实例

实例：controlCheck.html



（4）、表单验证实例

实例：loginFormCheck.html

### Javascript事件绑定

（1）、静态绑定

实例：staticBindEvent.html

<button type="button" id="btn1">绑定事件</button>  
  
<script type="text/javascript">  
var btn1= document.getElementById("btn1");   
btn1.onclick = function(){  
 alert("btn1单击事件绑定成功！");  
}

</script>

（2）、动态态绑定

实例：dynamicBindEvent.html

function addEvent(el, type, fn) {   
　　if(el.addEventListener){  
　　 el.addEventListener(type,fn,false); *//DOM2.0*

　　}  
 else if(el.attachEvent()){  
　　 el.attachEvent('on' + type,fn,false); *//IE5+*

　　}  
 else{  
　　 return false  
 }  
}

function append(){  
　　var body=document.getElementById('container');  
　　var btn=document.createElement('button');  
　　btn.type='button'

　　btn.innerHTML='确定'

*addEvent(btn,'click',function(){  
　　 console.log(this.tagName);　  
 }  
}   
window.onload=function(){* append(); *}*

<div id='container'></div>

### Javascript全局变量和局部变量

全局变量：不是放在函数内部的。

var.html

<**script**>  
 **id** = 1000; *// 全局变量，可以带var*   
 **function** *test*(){  
 **a** = 10; *// 没有var则是全局变量* **var** c = 20; *// 有var 则是局部变量  
 alert*(**'a1:'** + **a**);  
 **if** (**a** > 1)  
 {  
 **var** b = 20;  
 *alert*(**'b1:'** + b);  
 }  
 *alert*(**'b2:'** + b);  
 *alert*(**'c1:'** + c);  
 }  
 *test*();  
 *alert*(**'id:'** + **id**);  
 *alert*(**'a2:'** + **a**);  
 *alert*(**'b3:'** + b); *// 异常  
 alert*(**'c2:'** + c); *// 异常*</**script**>

### javascript运算符

（1）、算数运算符

假定 **y=5**

|  |  |  |  |
| --- | --- | --- | --- |
| **运算符** | **描述** | **例子** | **结果** |
| + | 加 | x=y+2 | x=7 |
| - | 减 | x=y-2 | x=3 |
| \* | 乘 | x=y\*2 | x=10 |
| / | 除 | x=y/2 | x=2.5 |
| % | 求余数 (保留整数) | x=y%2 | x=1 |
| ++ | 累加 | x=++y | x=6 |
| -- | 递减 | x=--y | x=4 |

（2）、赋值运算符

假定 **x=10** 和 **y=5**

|  |  |  |  |
| --- | --- | --- | --- |
| **运算符** | **例子** | **等价于** | **结果** |
| = | x=y |  | x=5 |
| += | x+=y | x=x+y | x=15 |
| -= | x-=y | x=x-y | x=5 |
| \*= | x\*=y | x=x\*y | x=50 |
| /= | x/=y | x=x/y | x=2 |
| %= | x%=y | x=x%y | x=0 |

（3）、比较运算符

假定 x=5：

|  |  |  |
| --- | --- | --- |
| **运算符** | **描述** | **例子** |
| == | 等于 | x==8 为 false |
| === | 全等（值和类型） | x===5 为 true；x==="5" 为 false |
| != | 不等于 | x!=8 为 true |
| > | 大于 | x>8 为 false |
| < | 小于 | x<8 为 true |
| >= | 大于或等于 | x>=8 为 false |
| <= | 小于或等于 |  |

（4）、逻辑运算符

假定 x=6 以及 y=3：

|  |  |  |
| --- | --- | --- |
| **运算符** | **描述** | **例子** |
| && | and | (x < 10 && y > 1) 为 true |
| || | or | (x==5 || y==5) 为 false |
| ! | not | !(x==y) 为 true |

### javascript语句

### 分支语句

（1）、if分支

if (条件)

{

当条件为 true 时执行的代码

}

{}中如果只有一行代码，括号可省略，但不推荐。

（2）、if else分支

if (条件)

{

当条件为 true 时执行的代码

}

else

{

当条件不为 true 时执行的代码

}

实例：ifelse.html

var x=new Date();

x.setFullYear(2100,0,14);

var today = new Date();

if (x>today){

    alert("今天是2100年1月14日之前");

}

else{

    alert("今天是2100年1月14日之后");

}

（3）、if elseif else分支

if (条件)

{

当条件为 true 时执行的代码

}

else if(条件)  
{  
 当条件为 true 时执行的代码

}

else

{

当条件不为 true 时执行的代码

}

中间的else if可以有多条。

实例：ifelseifelse.html

**dtNow** = **new *Date***();  
**hour** = **dtNow**.getHours();  
**if** (**hour** <= 12){  
 ***document***.write(**"上午好"**);  
}  
**else if** (**hour** > 12 && **hour** <= 18) {  
 ***document***.write(**"下午好"**);  
}  
**else**{  
 ***document***.write(**"晚上好"**);  
}

（4）、Switch分支

switch(n)

{

case 1:

执行代码块 1

break;

case 2:

执行代码块 2

break;

default:

n 与 case 1 和 case 2 不同时执行的代码

}

Default分支可以没有。

特别要注意的是break漏掉的情况。

var day=new Date().getDay();

switch (day)

{

case 0:

x="今天是星期日";

break;

case 1:

x="今天是星期一";

break;

case 2:

x="今天是星期二";

break;

case 3:

x="今天是星期三";

break;

case 4:

x="今天是星期四";

break;

case 5:

x="今天是星期五";

break;

case 6:

x="今天是星期六";

break;

}  
alert(x);

### 循环语句

（1）、for循环语句

语法：

for (初始语句1; 条件判断2; 改变条件值4)

{

被执行的代码块3

}

执行顺序1-2-3-4-2-3-4-2-…..

例子：

cn = 0;

for (var i=1; i<=10; i++)  
{  
 cn = cn + i;  
}  
alert(cn);

（2）、for in循环语句

var cars=new Array("Saab","Volvo","BMW");  
for (var item in cars){  
 *alert*(***cars***[***item***]);  
}

（3）、while循环语句

cars=["BMW","Volvo","Saab","Ford"];

var i=0;

while (cars[i]) {

document.write(cars[i] + "<br>");

i++;

}

（4）、do while循环语句

i = 0;  
do

{

document.write("The number is " + i + "<br>");

i++;

} while (i<5);

（5）、循环中断break, continue

实例：continue.html

### 条件判断语句

语法：

variablename=(condition)?value1:value2

例子：

greeting=(visitor=="PRES")?"Dear President ":"Dear ";

和下面的语句等价:

if (visitor=="PRES")

greeting="Dear President "

else

greeting="Dear ";

### javascript字符串的常用操作函数

常用方法

|  |  |
| --- | --- |
| **方法** | **描述** |
| [charAt()](http://www.w3school.com.cn/jsref/jsref_charAt.asp) | 返回在指定位置的字符。 |
| [concat()](http://www.w3school.com.cn/jsref/jsref_concat_string.asp) | 连接字符串。 |
| [indexOf()](http://www.w3school.com.cn/jsref/jsref_indexOf.asp) | 检索字符串。 |
| [lastIndexOf()](http://www.w3school.com.cn/jsref/jsref_lastIndexOf.asp) | 从后向前搜索字符串。 |
| [match()](http://www.w3school.com.cn/jsref/jsref_match.asp) | 找到一个或多个正则表达式的匹配。 |
| [replace()](http://www.w3school.com.cn/jsref/jsref_replace.asp) | 替换与正则表达式匹配的子串。 |
| [search()](http://www.w3school.com.cn/jsref/jsref_search.asp) | 检索与正则表达式相匹配的值。 |
| [slice()](http://www.w3school.com.cn/jsref/jsref_slice_string.asp) | 提取字符串的片断，并在新的字符串中返回被提取的部分。 |
| [split()](http://www.w3school.com.cn/jsref/jsref_split.asp) | 把字符串分割为字符串数组。 |
| [substr()](http://www.w3school.com.cn/jsref/jsref_substr.asp) | 从起始索引号提取字符串中指定数目的字符。 |
| [substring()](http://www.w3school.com.cn/jsref/jsref_substring.asp) | 提取字符串中两个指定的索引号之间的字符。 |
| [toLowerCase()](http://www.w3school.com.cn/jsref/jsref_toLowerCase.asp) | 把字符串转换为小写。 |
| [toUpperCase()](http://www.w3school.com.cn/jsref/jsref_toUpperCase.asp) | 把字符串转换为大写。 |

（1）、字符串的连接

var txt1="Hello";  
var txt2="world";  
var ret = txt1 + txt2;

alert(ret);

（1）、字符串的长度

var txt="Hello world!"

document.write(txt.length);

（2）、子字符串的位置

var str = "Hello world!";

var pos = str.indexOf("Hello");

（3）、替换字符串

stringObject.replace(regexp/substr,replacement)

var str="Visit Microsoft!";

document.write(str.replace("Microsoft", "W3School"));

（4）、字符串切分

var str="How are you doing today";

var arrStr = str.split(" ");  
for (var i = 0; i < arrStr.length; ++i) {  
 document.write(arrStr[i]);  
}

### javascrip浏览器对象模型（BOM）

（1）、window对象

获得浏览器的尺寸

var w=window.innerWidth  
|| document.documentElement.clientWidth  
|| document.body.clientWidth;  
  
var h=window.innerHeight  
|| document.documentElement.clientHeight  
|| document.body.clientHeight;

（2）、location对象

设置网址，可以将浏览器导航到到指定网址：

window.location.href="<http://www.baidu.com>";

（3）、history对象

window.history.back(); // window.history.go(-1);

实例：first.html、detail.html

<a onclick="*history*.back()" href="#">返回</a>

注意#不能少。

也可以用其他标签的单击事件来返回到上一个页面。

<span onclick="*history*.back()">返回</span>

window.history.forward();

（4）、navigator对象

实例：navigator.html

<div id="example"></div>  
<script>  
txt = "<p>浏览器代号: " + navigator.appCodeName + "</p>";  
txt+= "<p>浏览器名称: " + navigator.appName + "</p>";  
txt+= "<p>浏览器版本: " + navigator.appVersion + "</p>";  
txt+= "<p>启用Cookies: " + navigator.cookieEnabled + "</p>";  
txt+= "<p>硬件平台: " + navigator.platform + "</p>";  
txt+= "<p>用户代理: " + navigator.userAgent + "</p>";  
txt+= "<p>用户代理语言: " + navigator.systemLanguage + "</p>";  
document.getElementById("example").innerHTML=txt;  
</script>

### javascrip面向对象编程

（1）、使用字面值（literal）来创建对象

literal.html

**var *person***={  
 **"name"**:**"Jack"**,  
 **"age"**:30,  
 speak:**function** () {  
 *alert*(**'speak!'**);  
 }  
};

字面值法创建对象强调该对象仅是一个可变的hash映射，而不是从对象中提取的属性或方法。

对象字面量的值可以是任意数据类型。

访问字面值对象：

var *name* = *person*.name;  
*alert*(*name*);  
var *age* = *person*['age'];  
*alert*(*age*);  
*person*.speak();

对象字面量的值可以是任意数据类型：

var Swapper = {

// 数组字面量（用逗号分隔，所有都要加引号）

images: ["smile.gif", "grim.gif", "frown.gif", "bomb.gif"],

pos: { //嵌套对象字面量

x: 40,

y: 300

},

onSwap: function() { //函数

}

};

（2）、使用构造函数创建对象

function Cat(name,color){

　　　　this.name = name;

　　　　this.color = color;

}

Cat.prototype.type = "猫科动物";

Cat.prototype.eat = function(){alert("吃老鼠")};  
  
var cat1 = new Cat("大毛","黄色");

var cat2 = new Cat("二毛","黑色");

alert(cat1.type); // 猫科动物

cat1.eat(); // 吃老鼠

### javascrip操作cookie

cookie可以讲网站的一些信息保存到客户端，当再次打开网页时可以直接从客户端读取，而不必到服务端去获取信息，从而提供网站的性能。

实例：cookie.html