# Javascript 基础

console.log() 是在火狐浏览器控制台输出日志 ,ie浏览器默认不支持console.log()

alert() 弹出消息对话框，并且alert消息对话框通常用于一些对用户的提示信息

1. **JS与DOM的关系**

浏览器有渲染html代码的功能，吧html源码在内存里行程一个DOM对象，就是文档对象

浏览器内部有一个JS的**解释器/执行器/引擎**、如chrome 用V8引擎

我们在html里写一个JS代码，JS代码被引擎所执行，而执行的结果就是对DOM的操作

而对学习DOM操作的结果，就是我们常常看到的特效，比如图片漂浮，文字变色

学习Javascript要分清

1. js语言本身的语法
2. DOM对象（把body,div,p等节点树看成一个对象）
3. BOM对象（把浏览器的地址栏，历史记录，DOM等庄子啊一个对象）

浏览器是宿主，但JS的宿主不限于浏览器，也可以是服务器端

1. **JS如何引入**
2. 页内的script代码

<script>

code…….

</script>

1. 外部js文件

<script src=”js/jquery.js”></script>

注意：外部的.js文件里面直接写js代码

**不要在开头和结尾加标签**

1. 错误的引入方式

<script src=”xx.js”>

var \_name =’张三’；

alert(\_name)；

</script>

1. Script标签写在页面的哪个位置？

页面head和body都可以写

1. 多段script的执行顺序

按引入顺序，逐段执行（思考为什么N多页面吧JS写在最后？）

1. 防止页面一次出现，等待时间较长，不符合人的阅读习惯
2. 防止js语言已经出现，但是变量在js之后没有在内存中读取，导致js找不到变量发生错误
3. **变量声明**

**命名规范**

JS的变量名可以用\_,数字,字母,$,组成,且数字不能开头

声明变量用var变量名来声明

var a= 34;

var b=45;

alert(a+b);

var $ = ‘jquery’;

alert($);

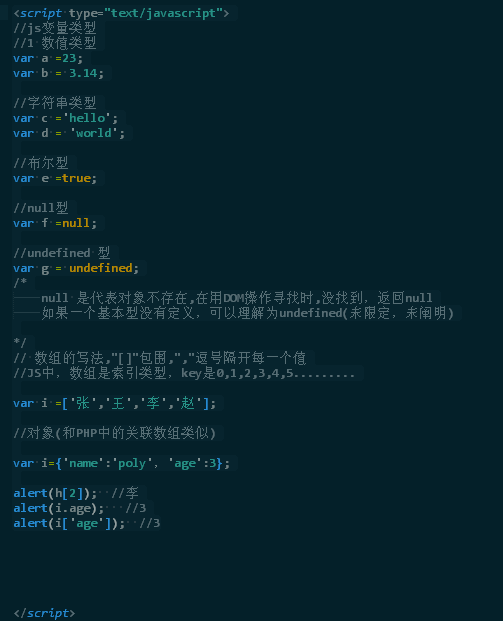
c=56;

alert(c);

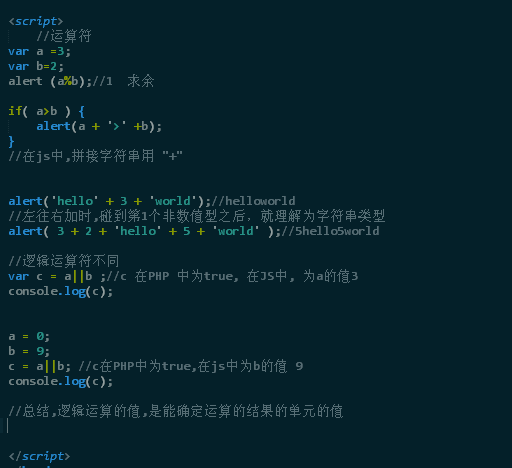
**注意**：变量名区分大小写str 和Str 不是一个变量

**注意**：不用var,会污染全局变量

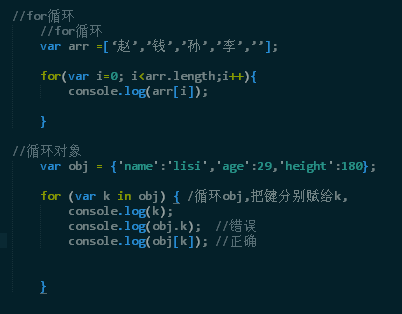
1. **JS变量类型**



1. **JS运算符**



1. **控制结构**



1. **对象操作**

String字符串对象

length属性:长度

concat(String)连接两个或多个字符串

indexOf(string)返回出现字符串的位置

substr(num1,[num2])截取字符串

toLowerCase()转换成小写

toUpperCase()转换成大写

replace(str1,str2)字符串替换

Date日期对象

getYear()返回年份(2位或4位)

getFullYear()返回年份(4位)

getMonth()返回月份0-11

getDate() 返回日期1-31

getDay() 返回星期数0-6

getHours() 返回小时数0-23

getMinutes() 返回分钟数0-59

getSeconds() 返回秒钟数0-59

getmilliseconds() 返回毫秒数 0-999

内置对象：

日起对象类：Date（）

var today = new Date();

Date() 函数是所有日期时间对象的一个“抽象模板”，而today是通过Date（）造出来的一个对象。

还有一个对象，是js已经创建完毕的，不必再new来得到。

Math对象

Math.random(); 返回0~1之间的随机数

Math.ceil(); 对数进行上舍入 //math.ceil(0.6) 返回1 math.ceil(-5.1) 返回-5

Math.foor(); 对数进行下舍入

Math.round; 把数四舍五入为最接近的整数

sqrt() 开平方根

pow(a,b) 返回数值a 的,数值b的次方

min(a,b) 返回最小值

max(a,b) 返回最大值

数组对象

concat() 返回一个由两个数组合并组成的新数组

join() 返回一个由数组中的所有元素并返回新长度

pop() 删除数组中的最后一个元素并返回该值

push() 像数组中添加新的元素,返回数组的新长度

shift() 删除数组中的第一个元素并返回该值

unshift() 返回一个数组,在数组头部插入了指定的元素

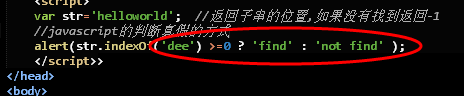
sort() 返回一个元素被排序了的array对象

reverse() 返回一个元素反序的array对象

slice() 返回数组的一个片段

splice() 从数组中删除元素

1. **JS内置对象**



indexOf 是找字符串返回位置

1. **浏览器window对象**

**注**：**window对象是浏览器宿主对象**,和JS语言无关 象

**window对象的方法**

window.alert(message) 类似js的alert

window.conflrm(message)

window.prompt(message[,defaultstr])

window.close() 关闭窗口

window.print() 打印

**window.setlntval (表达式,毫秒)**

**window.clearlntval(定时器对象)**

**window.setTimeout(表达式,毫秒)**

**window.clearTimeout(定时器对象)**

window对象的子对象:

**navigator** 浏览器信息对象

appCodeName 内部代码

appName 浏览器名字

appVersion 浏览器版本

platform 操作系统类型

userAgent 用户代理信息 头信息

cookieEnable 是否支持cookie

判断浏览器是否支持cookie

<http://www.zixue.it/thread-12911-1-1.html>

**location** 地址栏对象

host主机

port端口

pathname路径

protocol 协议

search 查询字符串

assign(url)页面跳转

**history** 历史记录

length: 历史记录的数目

back()

forword()

go()

screen屏幕对象

height 高度

width 宽度

availHeight 可用高度

availWidth 可用宽度

colorDepth 颜色

**document HTML文档对象**

即HTML代码形成的对象,操作此对象,可动态的改变页面的内容

是我们做JS的主战场

1. **作用域**

**var的本质:**

var a=3;//声明a变量并赋值

b=3;//只是一个单纯的”赋值”

JS作用域的特点:

首先在函数内部查找变量,找不到则到外层函数查找,逐步找到最外层

即window对象,并操作window对象的属性.

console.log(window.a,window.b);

function t(){

var a= ’local’;

b = ’global’;

}

console.log(window.a,window.b);

//a的值不变,b的值为global, 由于函数内部没有定义b

//找到外部变量b 并将gloal赋值于b

**思考 :** 为什么可以直接写alert(),setIntervall()? (setintervall())

1. **第一个简单JS效果**

<style>

#test1{

width:300px;

height:300px;

background:blue;

}

#test2 {

width:300px;

height:300px;

background:red;

}

</style>

</head>

<body>

<div id=”test1” onclick=”ch();”></div>

</body>

<script>

function ch(){

//1 要找到这个div对象

var test1 =document.getElementById(‘test1’);

//2 要修改它的属性

test1.id =’test2’;

}

</script>

</html>

1. **读取和改变对象属性**

**小练习：灯泡开关练习**



1. **找对象**



1. **操作对象的属性**



**img对象**

**.src**

**.alt**

**.title**

**style**

普通属性可以通过

对象.标签属性 来访问

width

height

borderTop

CSS属性通过

对象.style.属性

**例:**

**obj.alt**

**obj.title**

**obj.style.width**

**注意:**

标签属性与DOM对象属性的对应关系:

绝大部分两者是相同的,如:

imgobj.src属性对应<img src=””>中的src属性

但也有例外,如:

<div class=”main”>中,操作class属性用divobj.className

css属性与DOM对象属性的对应关系:

2者通过obj.style.css属性名对应

如:

obj.style.width

obj.sytle.background

如果css属性带有横线,如border-top-style,

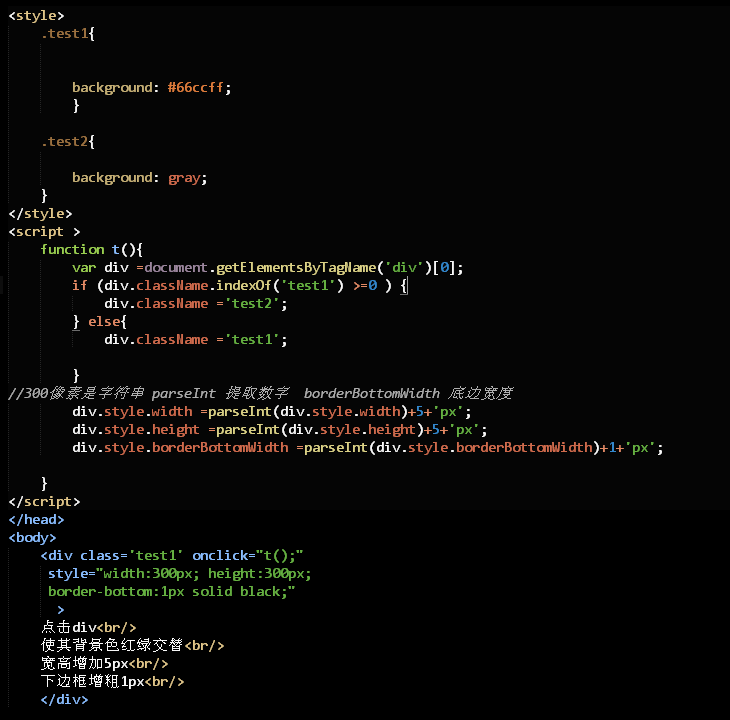
则把横线去掉后的字母大写

如:

obj.style.borderTopStyle

obj.style.marginLeft

小练习:点击div,使其宽,高加10px,底边增宽1px



1. **获取对象在内存中计算后的样式**

上节中，obj.style只能取得”内敛style”的值

对于<style></style>中的css属性值,则无能为力

我们可以用obj.currentStyle (ie内核) 和window.getComputedStyle(一般浏览器内核)来获取.

注意:只有IE和Opera支持使用currentStyle获取HTMLElement的计算后的样式,其他浏览器中不支持

标准浏览器中使用getComputedStyle,IE9以上也支持getComputedStyle.

window.getComputedStyle(obj,伪元素)

参数说明

第一个参数为要获取计算后的样式的目标元素

第二个参数为期望的伪元素,如:’after’,’frist-letter’等,一般为null

考虑兼容性,封装函数

function getStyle: function(el,attr){

return el.currentStyle?el.currentStyle[attr]:getComputedStyle(el,null)[attr];

}

注意：这2个方法，获取的对象是只读的，要改样式，还得靠obj.style

1. **删除对象**

**步骤:**

1. 找到对象
2. 找到他的父对象parenetObj

parentNode传回目前节点的父节点。只能应用在有父节点的节点中。

1. parentObj.removeChild(删除子对象)

**练习**：删除最后一个li

<ul>

<li>春</li>

<li>夏</li>

<li>秋</li>

<li>冬</li>



1. **创建对象**
2. 创建对象
3. 找到父对象 parentObj
4. parentObj.addChild(对象);

练习：添加1个li

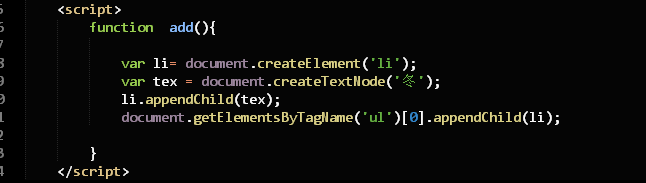
<ul>

<li>春</li>

<li>夏</li>

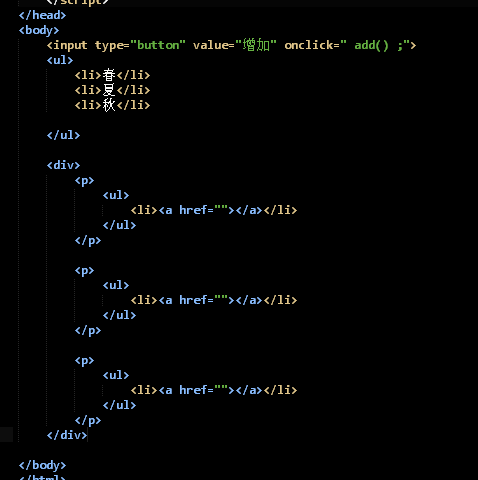
<li>秋</li>

<li>冬</li>

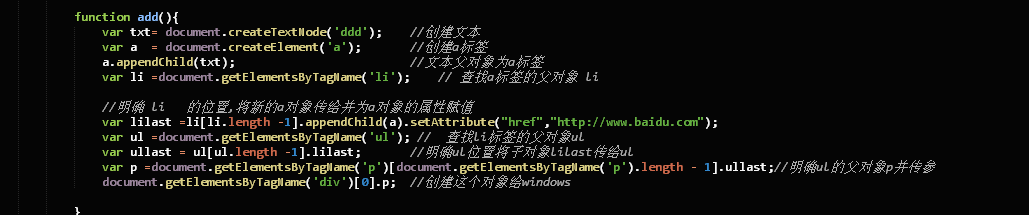


添加属性 函数 setAttribute 用法 obj.setAttribute(“href”,”#”);

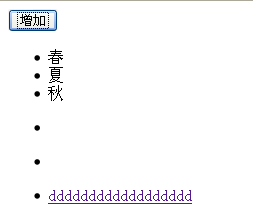
练习2



在最后一个a标签内添加内容和连接 点击一次添加一个



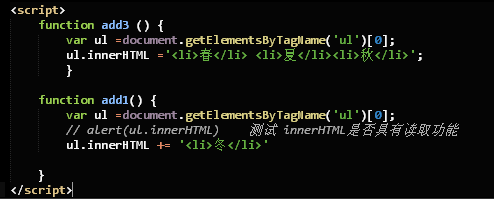
效果



1. **暴力操作节点**

**innerHTML:**代表节点内的内容，能读能写

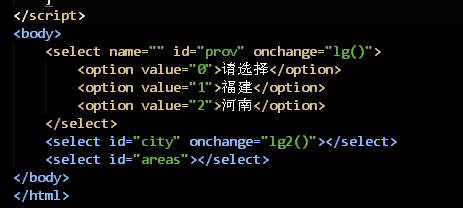
不是一个w3c规定的标准对象属性，但是-------各浏览器支持的很好



1. **联动菜单**

所用知识点：事件 +DOM 操作

三级菜单不是正规写法,只是尝试不建议用



1. **定时器**

**windows.setTimeout**(‘语句’,’毫秒’);指定毫秒后执行一次语句

注：定时器不属于js知识，他是windows对象提供的功能



**windows.setInterval**  (‘语句’,’毫秒’); //每经过执行语句

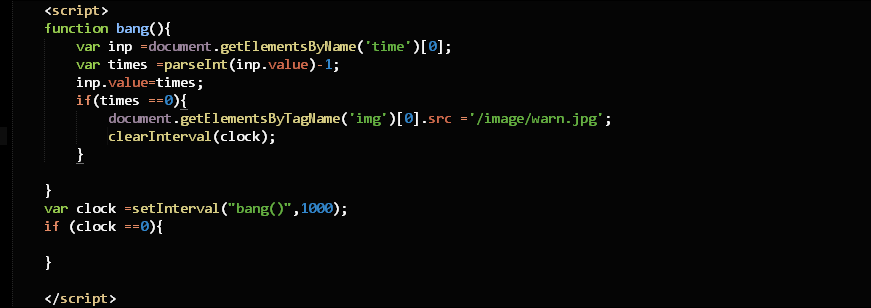
小练习:倒计时炸弹  
思路:每1秒,修改剩余时间,剩余时间为0时,修改爆炸后的图片



出现负值

**清除定时器:**

* clearInterval()
* clearTimeout()



**setTimeout实现倒计时**

setTimeout 可以反复调用自身，达到类似setInterval的效果

例：setTimeout版的定时炸弹



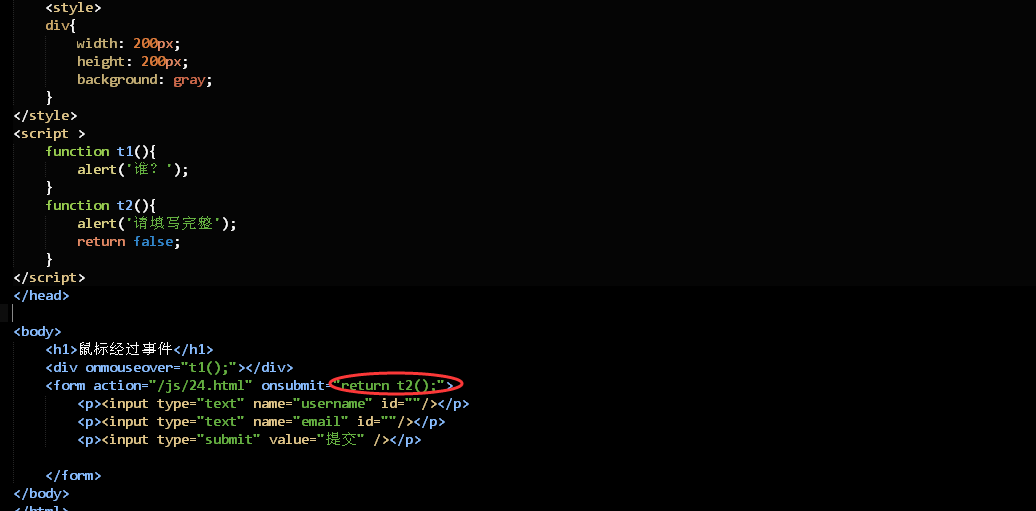
1. **DOM中的事件**

* onclick 元素点击时
* onfocus 元素获得焦点时
* onblur 元素失去焦点时



* onmouseover 鼠标经过时
* onsubmit 表单提交时
* onload 页面加载完毕时 运行

注意：onsubmit=”return t2()”;t2()函数才能拦截提交效果



1. **结构样式行为分离**



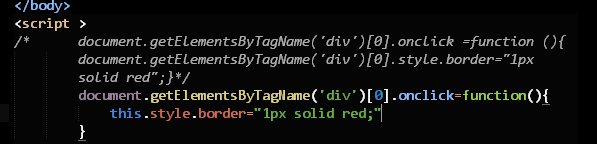
**结构 行为 样式**

结构行为样式分离



通过属性绑定方法

this指的就是div对象



**事件对象**Event 对象

**事件对象:**事件发生的瞬间,发生位置,时间,鼠标按键,触发的节点等信息,被打包成1个对象.

此对象,系统自动传递给时间函数的第1个参数.

**小练习:**



**事件委托**

当有比较多元素需要绑定某事件时，可以把事件绑定在他们的父元素上

把事件处理委托给了他的上级

重要属性target



1. **JS使用正则**

**声明**

var patt=/\*\d{11}$/;

**使用**

//判断String是事符合正则要求

patt.test(String);

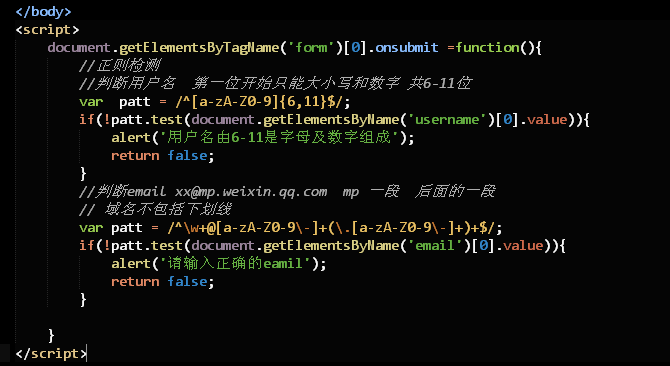
//找出字符串中的符合正则的子串

patt.exec(String);

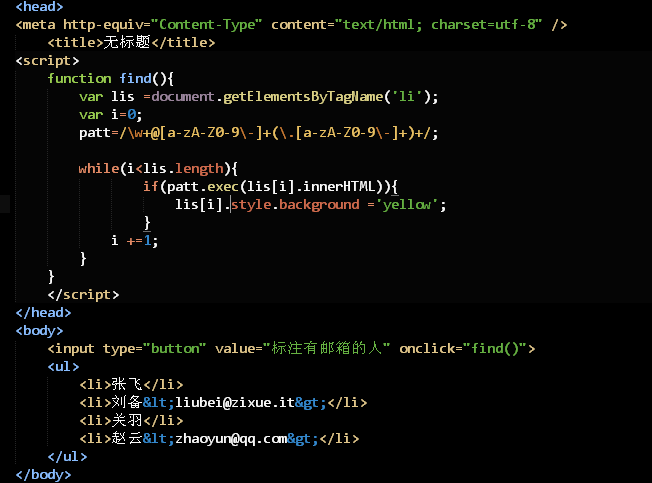
^ ：匹配输入的开始位置。  
\：将下一个字符标记为特殊字符或字面值。  
\* ：匹配前一个字符零次或几次。  
+ ：匹配前一个字符一次或多次。  
(pattern) 与模式匹配并记住匹配。  
x|y：匹配 x 或 y。  
[a-z] ：表示某个范围内的字符。与指定区间内的任何字符匹配。  
\w ：与任何单词字符匹配，包括下划线。

{n,m} 最少匹配 n 次且最多匹配 m 次  
$ ：匹配输入的结尾。

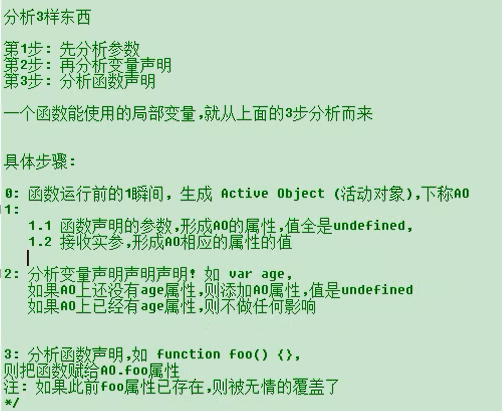
例子:



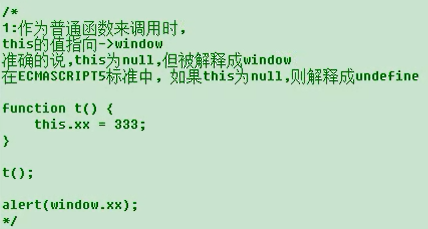
正则检测 patt.exec(String)



JS的词法分析



JS中函数的4种调用方式



JS面向对象的原型链

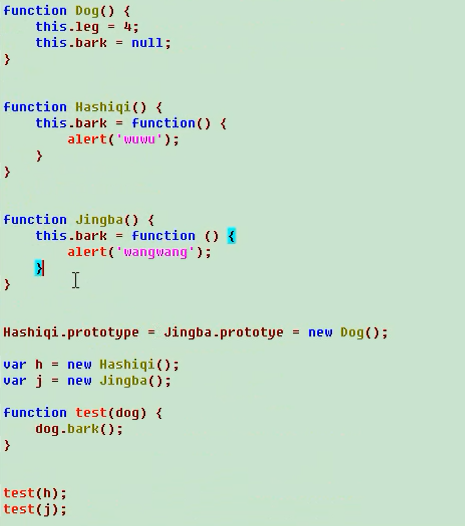


对象-->原型-->原型的原型-->Object对象-->NULL

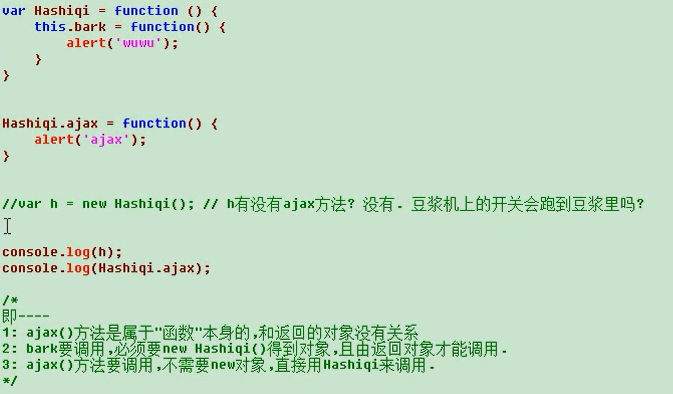
这样的一条链称为原型链，对象的属性和方法就是沿着原型链来查找和调用的

也就是JS中的原型继承

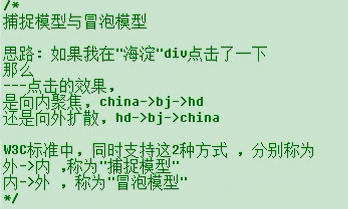
JS的多态，实际上在js里谈不上多态，因为js就没有限制参数类型



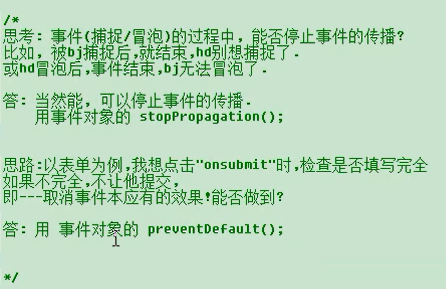
JS的静态方法，静态方法不需要new对象，可以通过函数名.方法名()直接调用



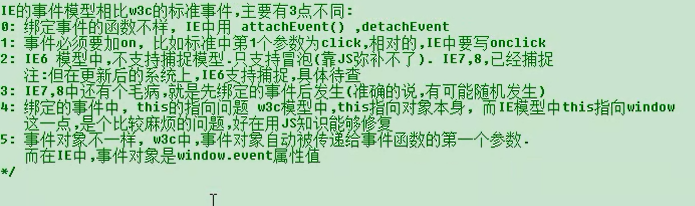
JS的捕捉模型和冒泡模型



阻止事件的效果



事件模型在IE低版本中的不同之处



Js中获取事件对象的方法详解：http://www.jb51.net/article/26552.htm

# JS:自定义兼容性强的事件函数模型-EventUtil

/\*element代表元素，type代表事件类型，handler代表事件被触发时调用的方法 /

var EventUtil = {

addHandler: function (element, type, handler) {

        if (element.addEventListener) {

            element.addEventListener(type, handler, false);

        } else if (element.attachEvent) {

            element.attachEvent("on" + type, handler);

        } else {

            element["on" + type] = handler;

        }

    },

    preventDefault: function (event) {

        if (event.preventDefault) {

            event.preventDefault();

        } else {

            event.returnValue = false;

        }

    }

};

使用方法：EventUtil.addHandler(element, type, handler);