

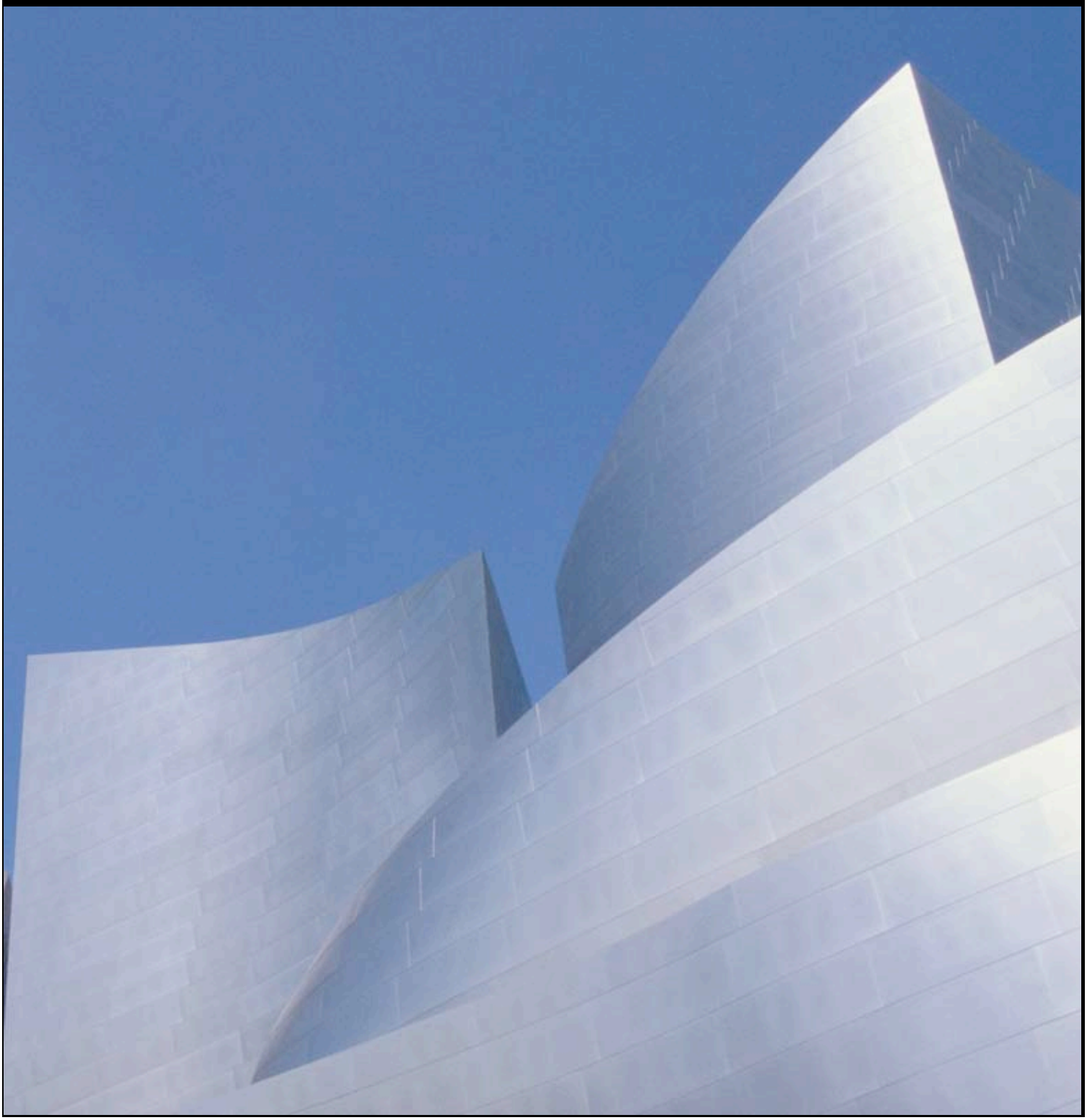
http://www.xaygc.com

云工厂-西安 Java 培训第一选择

JavaScript

旺旺老师简介

刘红旺，82 年生于山西，02 年来西安求学，一年后辍学从事软件开发工作，三年后投身软件教育至今。至 09 年 12 月，旺旺拥有七年工作经验，四年软件教育经验。旺旺兴趣广泛，近期一直学习经济学与心理学。旺旺老师课程特点：通俗易懂，风趣幽默。



图书说明

城市：陕西，西安

QQ： 22713528

技术交流群：17951956

电子邮件：shudaizi@163.com

个人网站：

http://www.xaygc.com

西安云工厂：专业信息技术服务商，
西安 Java 培训第一选择

说明：本教程配套视频会同期发布，欢迎大家访问旺旺老师的网站 http://www.xaygc.com 了解详情。

旺旺一直有逛书店的习惯，到 java 柜台总会停下来看看有什么基础书籍向学生推荐，但可惜十分满意的基本没有。于是有了下面的话：

现在市面上的 java 教材有两种：

一种是学院派老师编写的，他们是主流，你看到十本 java 书，有九本半是这样的。他们的作者拥有让人敬仰的称谓，如某某教授，某某专家；他们写的书大都是大部头，动辄上千上万；他们文风严谨，遣词专业；他们案例深奥，让人深思。总之，我很敬仰他们，因为我当初也是读着这样的书学 java 的。但现在看来，这样的书并不适合入门，情节大都是这样的，当我们怀着莫大的热情准备开始学习 java,买了一本久仰的《java 编程宝典》（有无此书，无从得之，是旺旺杜撰的书名），它很厚很重很专业，你用毅力恒心支持看了几十页，才相信，java 的确是难学，后来，就不怎么看它了。直到过了很多天，也许你已从事 java 方面的开发工作，一天，从书堆里无意的见到它，拍拍灰尘看看，够厚，就作为工具书备用吧。

还有一种是像旺旺这样草根阶层（好听点叫实战派）编写的，这样的书凤毛麟角，因为大家都在忙于工作，不像旺旺这样打了鸡血精力充沛的无法发泄。他们的称谓一般都不匝地，不谈也罢；他们的书也不怎么厚，能上千页的更少之又少；他们文风朴实，贴近大众；他们案例生动，通俗易懂，风趣幽默；你可以当技术类书籍来学习它，甚至当消遣类散文来阅读它。OK,如果你初学 java,需要的就是一本这样的书，所以**《旺旺老师 JavaSE 教程》的目标人群是：初学 java 的读者**。

还有如果您已认真完善的学习过 java,或已从事 java 方面的开发工作，那也可以看看本书的部分章节，旺旺一些幽默风趣的案例也许能给您带来一些惊喜,一些实用技术的深入讲解能给您带来些许收获。

所以《旺旺老师 JavaSE 教程》的**理想目标是： 让新手当技术类书籍来学习它，让老手当消遣类散文来阅读它**。想让所有人，在快乐中学习，在快乐中成长，在快乐中工作。学习 java 可以是件很轻松的事情，让所谓高深的技术平民化，让所谓复杂的技术简单化，这是我的奋斗目标,也会一直为此努力。

JavaScript

1.1 简介

SUN 与网景发明的，基于对象的，区分大小写的，弱的脚本语言。

vbScript(ms 80%)—>liveScript(功能比 vbScript 强大)

95 年 Java 出现火了

JavaScript(W3C)

JScript

基于对象:js 中虽然已经有类和对象的概念了，但又不完全满足面向对象的三个特征，所以把它称为基于对象的编程语言。

弱语言：JS 是弱语言，比如可以不用声明变量而直接使用。强语言如 C,C++,Java,C#。

外部 js:

<script language=“JavaScript” src= “文件名.js”>

</script>

注意：结束不能在一个标记中直接使用/>，而要单独列出</script>

为了 SEO,请把 js 内部放在最下面。

1.2 基本语法

1.2.1 变量与运算符

声明变量全部使用 var

1.2.2 函数

一般普通语言方法

```
返回类型（void） 方法名(参数类型 参数名, 参数类型 参数名){
        return 返回类型变量;
}
//声明不需要返回值
function functionName(a, b){//声明变量前没有数据类型
        return 变量;//有返回值直接返回，没有的话啥都不写
```

}

1.2.3 流程控制

1.2.4 常用函数

- parseInt (String) 把一个 String 转换为 int 类型
- parseFloat(String) 把一个 String 转换为 float 类型
- isNaN(String) is not a number 判断一个字符串是否不是数字，不是返回 true,是返回 false
- alert(msg)
- document.write(str);

1.2.5 案例计算器

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<meta http-equiv="description" content="JavaScript 计算器，云工厂旺旺老师开发"/>
<title>JavaScript 计算器-云工厂旺旺老师</title>
<script type="text/javascript">
	function getResult(type) {
		if(checkData(type) == false) {
			return;
		}
		var num1 = parseFloat(document.jisuanqi.num1.value);
		var num2 = parseFloat(document.jisuanqi.num2.value);
		var result;
		switch (type) {
			case '+':
				result = num1 + num2;
			break;
			case '-':
				result = num1 - num2;
```

```
                break;
                case '*':
                        result = num1 * num2;
                break;
                case '/':
                        result = num1 / num2;
                        break;
                case '%':
                        result = num1 % num2;
                break;

        }
        document.jisuanqi.result.value = result;
}
function checkData(type) {
        if(document.jisuanqi.num1.value == "") {
                alert("第一个操作数不能为空");
                return false;
        }
        if(document.jisuanqi.num2.value == "") {
                alert("第二个操作数不能为空");
                return false;
        }
        if(isNaN(document.jisuanqi.num1.value)) {
                alert("第一个操作数不是数字");
                return false;
        }
        if(isNaN(document.jisuanqi.num2.value)) {
                alert("第二个操作数不是数字");
```

```
                return false;
            }
            if(type == "/" && (document.jisuanqi.num2.value == 0)) {
                alert("除数不能为 0");
                return false;
            }
            return true;
        }
</script>
</head>
<body>
<p align="center">
        <form name="jisuanqi" />
            num1:<input name="num1"><br>
            num2:<input name="num2"><br>
            rslt:<input name="result" readonly><br>
            <input type="button" value="+" onclick="getResult('+')"/>
            <input type="button" value="-" onclick="getResult('-')"/>
            <input type="button" value="*" onclick="getResult('*')"/>
            <input type="button" value="/" onclick="getResult('/')"/>
            <input type="button" value="%" onclick="getResult('%')"/>
        </form>
</p>
</body>
</html>
```

1.3 调试工具

1.3.1 FIREBUG

Firebug 是一套与 Firefox 集成在一起的功能强大的 web 开发工具，可以实时编辑、调试和监测任何页面的 CSS、HTML 和 JavaScript。

扩展文件通常是 xpi 文件格式(以.xpi 结尾).有三种安装方法:

1,直接拖拽到 Firefox 浏览器的窗口里或扩展窗口里

2,点击 ’ 菜单"-"工具"-”扩展”,把.xpi 文件拖进弹出窗口里,稍后就可以看到 Firefox 会询问你是否要安装这个插件,点击是,并重启 Firefox 浏览器

3,鼠标右键点 xpi 文件，选打开方式，在打开方式里选 firefox，设置关联。以后只要双击 xpi 文件就自动安装了

断点快捷键：

F8:执行结束或者下个断点

F10:单步执行

F11:进入方法

1.3.2 IE 自带调试工具

F10:单步执行

F11:进入方法

1.4 常用对象

1.4.1 字符串 STRING

1.4.1.1 属性

- length ： 返回字符串的长度

1.4.1.2 方法

查找

- charAt(下标) 查找某个位置的字符
- indexOf(“字符串”) 字符串在当前字符串中第一次出现位置,没找到返回-1
- indexOf(“字符串”，index) 从 index 起字符串在当前字符串中第一次出现位置
- lastIndexOf()

截取

- substring(a, b) 从 a 开始，截取(b-a)个,与 java 相同
- substr (a, b) 从 a 开始，截取 b 个

- substr (a) 从 a 开始，到最后一个
- replace(“a”, “b”)；查找替换，只替换第一个，java 替换所有有
- split(“a”)按照某种规则把一个字符串切割成字符数组

其它：

- big( ) 增大字符串文本
- blink( ) 使字符串文本闪烁（IE 浏览器不支持）
- bold( ) 加粗字符串文本
- fontcolor( ) 确定字体颜色
- italics( ) 用斜体显示字符串
- strike( ) 显示加删除线的文本
- sub( ) 将文本显示为下标
- toLowerCase( ) 将字符串转换成小写
- toUpperCase( ) *将字符串转换成大写*

1.4.2 日期 DATE

Date 对象存储的日期为自 1970 年 1 月 1 日 00:00:00 以来的毫秒数

var 日期对象 = new Date (年、月、日等参数)

例如：

```
var  mydate=new Date("July 29, 2007,10:30:00" )
```

如果没有参数，表示当前日期和时间

例如：

```
var today = new Date()
```

方法：

setXxx 这些方法用于设置时间和日期值

getXxx 这些方法用于获取时间和日期值

用作 Date 方法的参数的整数范围：

Seconds 和 minutes 0 至 59

Hours 0 至 23

Day 0 至 6（星期几）

Month 0 至 11（一月至十二月）

Date 1 至 31（月份中的天数）

案例一：

```
<HTML>
<BODY>
<script language="javaScript">
var now= new Date( );
var hour = now.getHours( );
if (hour>=0 && hour <=12)
      document.write("上午好!")
if (hour>12 && hour<= 18)
    document.write("下午好!");
if (hour>18 && hour <24)
    document.write("晚上好!");
document.write("<P>今天日期:"+now.getYear()+"年 “
      +(now.getMonth( )+1)+"月"+now.getDate()+"日");
document.write("<P>现在时间:"+now.getHours()+"点"+now.getMinutes( )+"分");
</script>
</body>
</HTML>
```

案例一：跳动的时钟

```
<SCRIPT language="JavaScript">
function disptime( ){
 var time = new Date( ); //获得当前时间
 var hour = time.getHours( );  //获得小时、分钟、秒
 var minute = time.getMinutes( );
 var second = time.getSeconds( );
 var apm="AM"; //默认显示上午：AM
 if (hour>12) //按 12 小时制显示 {
    hour=hour-12;
    apm="PM"  ;
 }
 if (minute < 10) //如果分钟只有 1 位，补 0 显示
   minute="0"+minute;
 if (second < 10) //如果秒数只有 1 位，补 0 显示
   second="0"+second;
document.myform.myclock.value= hour+":"+minute+":"+ second+" "+apm;
var myTime = setTimeout("disptime(  )",1000);
}
</SCRIPT>
```

```
<STYLE type="text/css">
<!--
/*设置样式：无边框的文本框*/
INPUT {
```

```
        font-size: 50px;
        color: #CC0000;
        border-style:none
}
-->
</STYLE>
</HEAD>
<BODY onLoad="disptime( )">
<FORM NAME="myform">
<INPUT name="myclock" type="text"   value="" >
</FORM>
```

1.4.3 数组

var 数组名 = new Array(数组大小);

var emp=new Array(“AA”，“BB”，“CC”);

常用属性

length ：返回数组中元素的个数

常用方法

- join 将数组中的元素组合成字符串
- reverse 颠倒数组元素的顺序，使第一个元素成为最后一个，而最后一个元素成为第一个
- sort 对数组元素进行排序

1.5 内置对象

不需要声明创建，可以直接使用，内置对象全部为 js 关键字

1.5.1 WINDOW 对象

可以直接使用属性名或方法名，而不用写对象名。alert() parseInt() isNaN()

alert()本质上是 window.alert()

名称	说明
status	指定浏览器状态栏中显示的临时消息
screen	有关客户端的屏幕和显示性能的信息。
history	有关客户访问过的 URL 的信息。
location	有关当前 URL 的信息。
document	表示浏览器窗口中的 HTML 文档

名称	说明
alert ("提示信息")	显示一个带有提示信息和确定按钮的对话框
confirm("提示信息 “)	显示一个带有提示信息、确定和取消按钮的对话框
open ("url","name")	打开具有指定名称的新窗口
close ( )	关闭当前窗口
showModalDialog( )	在一个模式窗口中显示指定的 HTML 文档
setTimeout("函数",毫秒数)	设置定时器：经过指定毫秒值后执行某个函数

以下为用 java 类模拟 window 对象。

```
package com.wangwang;

public class Window {
	public Document document = new Document();
	public History  history  = new History();
	public Location location = new Location();
	public Screen screen = new Screen();
	public String status;

	/**
	 * 把一个字符串转换为一个整数
	 * @param arg
	 * @return
	 */
	public int parseInt(String arg) {
		return Integer.parseInt(arg);
	}
	/**
	 * 把字符串转换为float
	 * @param arg
	 * @return
	 */
	public float parseFloat(String arg) {
		return Float.parseFloat(arg);
	}
	/**
	 * 显示一个消息框
	 * @param arg 要显示的内容
	 */
```

```
	public void alert(String arg) {
		//msgbox()
	}
	/**
	 * 判断一个字符串是不是不是数字
	 * @param arg
	 * @return
	 */
	public boolean isNaN(String arg) {
		return false;
	}
	/**
	 * 显示一个问题对话框
	 * @param msg
	 * @return
	 */
	public boolean confirm(String msg) {
		return false;
	}
	/**
	 * 关闭当前窗体
	 *
	 */
	public void close() {

	}
	/**
	 * 打开一个新的窗体
	 * @param url
	 */
	public void open(String url) {

	}
	/**
	 * status
	height： 窗口高度；
	width： 窗口宽度；
	top： 窗口距离屏幕上方的象素值；
	left：窗口距离屏幕左侧的象素值；
	toolbar： 是否显示工具栏，yes为显示；
	menubar，scrollbars 表示菜单栏和滚动栏。
	resizable： 是否允许改变窗口大小，yes或1为允许
	location： 是否显示地址栏，yes或1为允许
	status：是否显示状态栏内的信息，yes或1为允许；

	toolbars=1, location=0
	 */
	public void open(String url, String name, String status) {

	}
	/**
	 * 显示模式窗体
	 * @param url 要显示的窗体的路径
	 */
	public void showModalDialog(String url) {

	}
}
```

1.5.2 SCREEN 对象

```
if((window.screen.width) != 1024 || (screen.height != 768)) {
      alert("请把你的显示器分辨率设置为1024*768以获得最好的显示效果");
}
```

1.5.3 LOCATION 对象

window.location.href = “目标 url”

window.location = = “目标 url”

location.href = “目标 url

可以简写为 location = “目标 url”

1.5.4 HISTORY 对象

back()

forward()

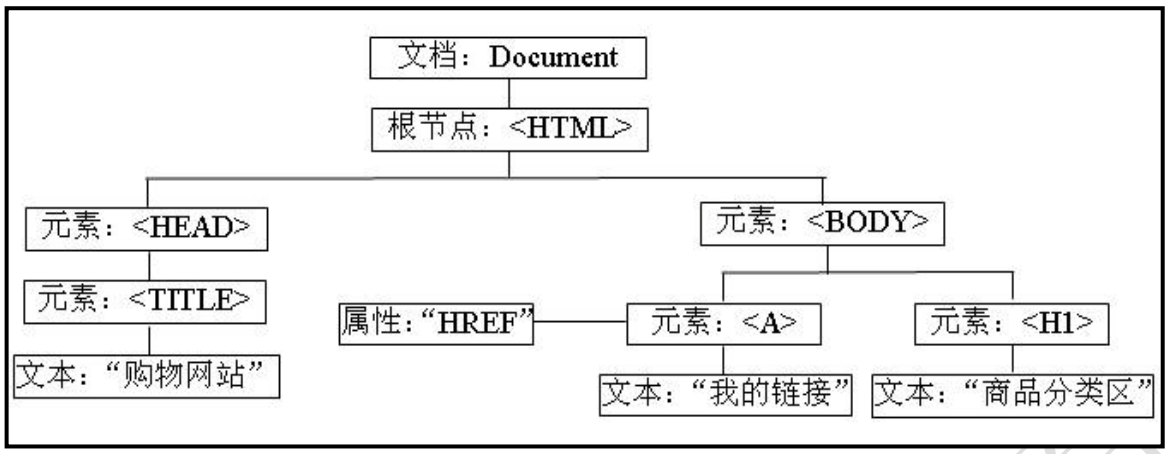
go(整数)负数表示 back

1.5.5 DOCUMENT 对象

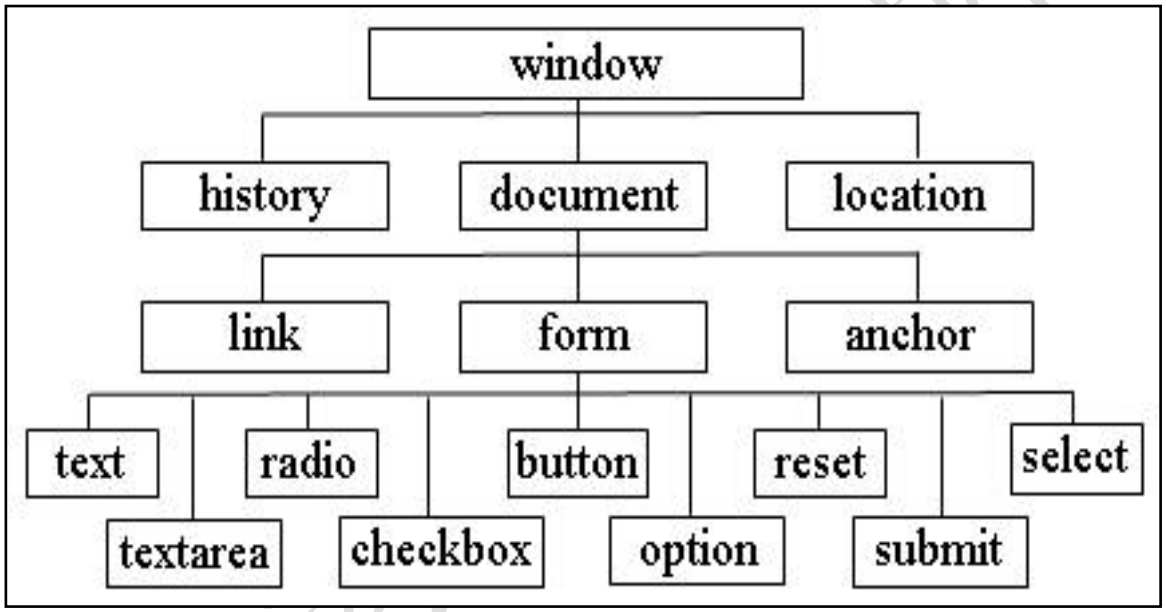
名称	说明
write()	往浏览器窗体 body 写入内容
getElementById()	根据节点 id 得到首个节点句柄（引用）
getElementsByName()	根据节点 name 得到句柄，返回数组
getElementsByTagName()	根据节点类别得到节点句柄
createElement(“ElementName”)	创建一个新节点
createTextNode(“文本”)	创建一个文本节点

1.6 DOM 编程

Document Object Model 文档对象模型，可以把整个 html 文档理解为一棵 dom 树。



DOM 编程就是为了动态(在代码运行之后)的给文档（DOM 树）添加一些节点或删除一些节点，或者得到修改节点的属性。



1.6.1 NODE 对象

可以把 DOM 中的每个节点理解为是一个类，它们有一个共同的父类 Node，类似 Java 中所有类默认继承 Object，DOM 中的每个节点都继承之 Node。如 img 标签 button 标签 for 标签(类)等都共同继承自 Node 类

根据子类拥有父类的属性和方法，只要弄懂 Node 类的属性和方法，就可以操作所有 html 节点的属性和方法。

表格一：node 节点的常用属性

名称	返回值	作用

documentElement	节点	获取文档的根节点
attributes	数组	如是 Element，以 nameNodeMap 返回属性
childNodes	节点数组	以 Node[]形式存放子节点
firstChild	节点	获取第一个子节点，没有返回 null
lastChild	节点	获取最后一个子节点
nextSibling	节点	获取后一个兄弟节点
previousSibling	节点	获取前一个兄弟节点
parentNode	节点	获取父节点
nodeType	整型	获取节点类型：1 为元素，2 为属性，3 为文本
nodeName	字符串	获取节点名称：如果节点为元素，返回元素
nodeValue	字符串	获取节点值

表格二：node 节点的类型即 nodeType 值

接口	nodeType 常量	值	备注
Element	Node.ELEMENT_NODE	1	元素节点
Attr	Node.ATTRIBUTE _NODE	2	节点属性
Text	Node.TEXT_NODE	3	文本节点
Comment	Node.COMMENT_NODE	8	注释文本
Document	Node.DOCUMENT_NODE	9	document
DocumentFragment	Node.DOCUMENT_FRAGMENT	11	Document 片段

表格三：node 节点的常用方法

名称	作用	参数
appendChild	追加子节点	子节点对象
setAttribute	设置属性的值	属性名， 值
removeChild	删除子节点	子节点对象
removeAttribute	删除属性	属性名称
replaceChild	替换子节点	新节点，老节点
createElement	创建元素节点	元素名称
createTextNode	创建文本节点	文本内容

insertBefore	插入节点。如该节点存在，则删除再插入它的位置	节点对象
cloneNode()	复制节点	是否包括子节点
hasChildNodes()	是否有子节点	无

1.6.2 EVENT

event 是事件对象，也是内置对象，表示发生的某个事件。

keyCode 当前的按键编码

srcElement 在那个控件中产生的此事件

1.6.3 重要知识点

this 写在某个过程内，如果这个过程是动态生成控件调用的，this 表示此动态控件，如果是已有控件，this 表示 window。

id 是为客户端编程服务，客户端通过 id 得到对象句柄，应用，服务器端通过 name 得到数值。

1.6.4 获取修改属性

只要在 html 标签有的属性值，我们就可以通过句柄变量名.属性名的形式访问。

1.6.5 动态添加删除节点 DOM 实现

见案例：动态添加删除节点 DOM 实现.html

删除：得到自己的父亲，再删除自己

```
document.createTextNode("文本")

document.createElement("td")

d2.appendChild(mya);

var deltr =this.parentNode.parentNode;
```

```
deltr.parentNode.removeChild(deltr);
```

1.6.6 DOM 编程的“潜规则”

一般我们创建 DOM 对象使用 document.createElement()与 document.createTextNode()，但是有两个对象比较特殊，可以使用 new 的方式创建。

```
var op = new Option();

var img = new Image();

父标记.insertBefore(新节点, 旧节点);
```

1.6.7 DOM 操作 SELECT

1.6.7.1 或得数值的两种方法

1，se.value select 的 value 就是某个选中的 option 的 value

2，se.options[se.selectedIndex].value

se.selectedIndex 表示当前选中元素的下标 从 0 开始

1.6.7.2 OPTION

var op = new Option(“text”,”value”);

op.text = “”;

op.value = “”

1.6.7.3 清空 SELECT 控件中所有数值

se.length = 0;

1.6.8 INNERHTML

使用 innerHTML 也可以动态的（在代码运行时）个 html 添加删除节点内容。

innerHTML 会解析运行其中的 html 代码，而 innerText 不会解析，整个内容原样显示。

innerHTML 与 createElement appendChild 这些 dom 编程方式比较：优点是简单，代码较少。缺点是没有后者灵活，功能强大。

最好的结果是两种方式配合使用。

1.7 FORM 表单验证

1.7.1 如何调用验证方法

第一：在 submit 按钮的 onclick 事件中添加，如果验证方法返回 true,表单提交，如果返回 false 表单不提交 onclick="return checkData();图像提交域本质与提交按钮相同

第二：在 form 表单的 onSubmit 事件中，只要使用提交按钮或图像提交域提交，onSubmit 事件一定会触发。如果验证方法返回 true,表单提交，如果返回 false 表单不提交 onsubmit="return checkData();"

第三：使用非提交按钮或非提交域提交表单，onSubmit()事件并不会触发，直接在调用 submit 方法的代码中作验证。要注意：form 表单的 submit 按钮的 name 属性不能是 submit,否则无法提交

```
function mySubmit(){
        if(checkData()) {
                var myf = document.getElementById("gaga");
                myf.submit();
        }
}
```

1.7.2 具体验证实现

思路：在 text 的失去焦点事件中触发某些方法，改变其后显示消息的 div（span）的背景颜色或者文字颜色,还有文字提示内容。

1.8 JQUERY 基础

1.8.1 什么是 JQUERY

jQuery 是一个优秀的 JavaScript 库。它由 John Resig 创建于 2006 年 1 月。它的体积很小，简化了 JavaScript 的各种操作，使遍历 HTML 文档、操作 DOM、处理事件、执行动画以及执行 Ajax 的操作，JQuery 的口号是“write less，do more（写更少的代码，做更多的事情）。

DWR 框架侧重于 JavaScript 与 Java 的交互， JQuery 框架则侧重于 JavaScript 本身。

1.8.2 JQUERY 优点

轻量级，强大的选择器，出色的 DOM 操作的封装，可靠的事件处理机制，完善的 Ajax 支持，出色的浏览器兼容性，行为层与结构层的分离，丰富的插件支持，完善的文档，开源

1.8.3 下载

JQuery 将核心部分分为 Production（生产模式，经过压缩的 JS 文件）和 Development（开发模式，未经压缩的 JS 文件）两种，两种功能相同，前者体积小巧，后者利于调式；UI 组件可以选择下载全部组件或部分组件，下载的组件中自带了默认的主题。

1.8.4 HELLOWORLD

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<script type="text/javascript" src="jquery-1.4.4.js"></script>
<script type="text/javascript">
	$(function(){
		$('input').click(function(){
			$('#hello').toggle('blind');
		});
	});
</script>
<title>hello Jquery</title>
</head>

<body>
<input type="button" value="Hello JQuery"/>
<div id="hello" style="background-color:red;  width:80%">
	<p>hello jquery</p>
</div>
</body>
```

```
</html>
```

1.8.5 JQUERY UI 与 JQUERY 插件

jQuery 本身开源，支持插件扩展它的新功能。我们也可以自己开发 JQuery 插件。

jQuery UI 实际上是 jQuery 插件，专指由 jQuery 官方维护的 UI 方向的插件。jQuery 本身注重于后台，没有漂亮的界面，而 jQuery UI 则补充了前者的不足，它是利用 jQuery 的扩展性设计的插件。提供了一些常用的界面元素，诸如对话框、拖动行为、改变大小行为等等。jQuery UI 使 JQuery 既有强大的后台，又有华丽的前台.。

jQuery UI 主要分为 3 个部分：交互、微件和效果库。

交互：交互部件是一些与鼠标交互相关的内容，包括拖拽,鼠标按下放下,改变大小等。

微件：主要是一些界面的扩展，包括对话框，滚动条，日期控件等，新版本的 UI 将包含更多的微件。

效果库：用于提供丰富的动画效果，让动画不再局限于 jQuery 的 animate()方法。

1.9 JQUERY 核心方法

得到对象句柄
操作属性（包括样式等）
添加删除节点

JQuery 中核心方法为 jQuery()或$()，两者功能相同，有四种用法，下面一一讲解：

1.9.1 查找元素（选择器）

在JavaScript编程中，查找出指定元素然后进行操作是很常见的编程模式，通过DOM方式查找元素往往使用getElementById或getElementsByTagName，如果元素不能根据id或标签名这样显著的特征查找（如查找所有的单选框），则更加麻烦。使用JQuery可以根据表达式快速的查找到特定元素，语法如下：

jQuery(表达式, [内容])或$(表达式, [内容])，返回 JQuery 对象。

JQuery代码	JavaScript代码
$('#id')	getElementById('id')
$('DIV')	getElementsByTagName('DIV')

1.9.2 替代 ONLOAD()方法

我们经常希望在页面加载的时候执行一段 JavaScript 代码，传统的方式是在 BODY 的 onload 事件中编写代码。但 BODY 的 onload 事件会在页面的全部内容（包括页面引用的图片）加载完后才执行，具有一定的滞后性，并且不能保证多个方法都调用。使用 JQuery 提供的方法能够在页面的 HTML 代码加载完后就执行代码，语法如下：

```
<script type="text/javascript">
	/*
	window.onload = function(){
			alert("window.onload 第一次使用<br>");
	};
	window.onload = function(){
			alert("window.onload 第二次使用<br>");
	};
	*/
	jQuery(document).ready(function(){
			alert("window.onload 第一次使用<br>");
	});
	$(document).ready(function(){
			alert("window.onload 第二次使用<br>");

	});
	$().ready(function(){
			alert("window.onload 第三次使用<br>");
	});
	$(function(){
		alert("window.onload 第四次使用<br>");
	});
```

```
</script>
```

1.9.3 查找元素

1.9.4 查找元素

其它

Object -> Element

```
$("#divId")[0]
```

Element -> Object

```
$(document.getElementById(sID));
```


版本	修改内容	时间
V1.0	创建	2010－09－04