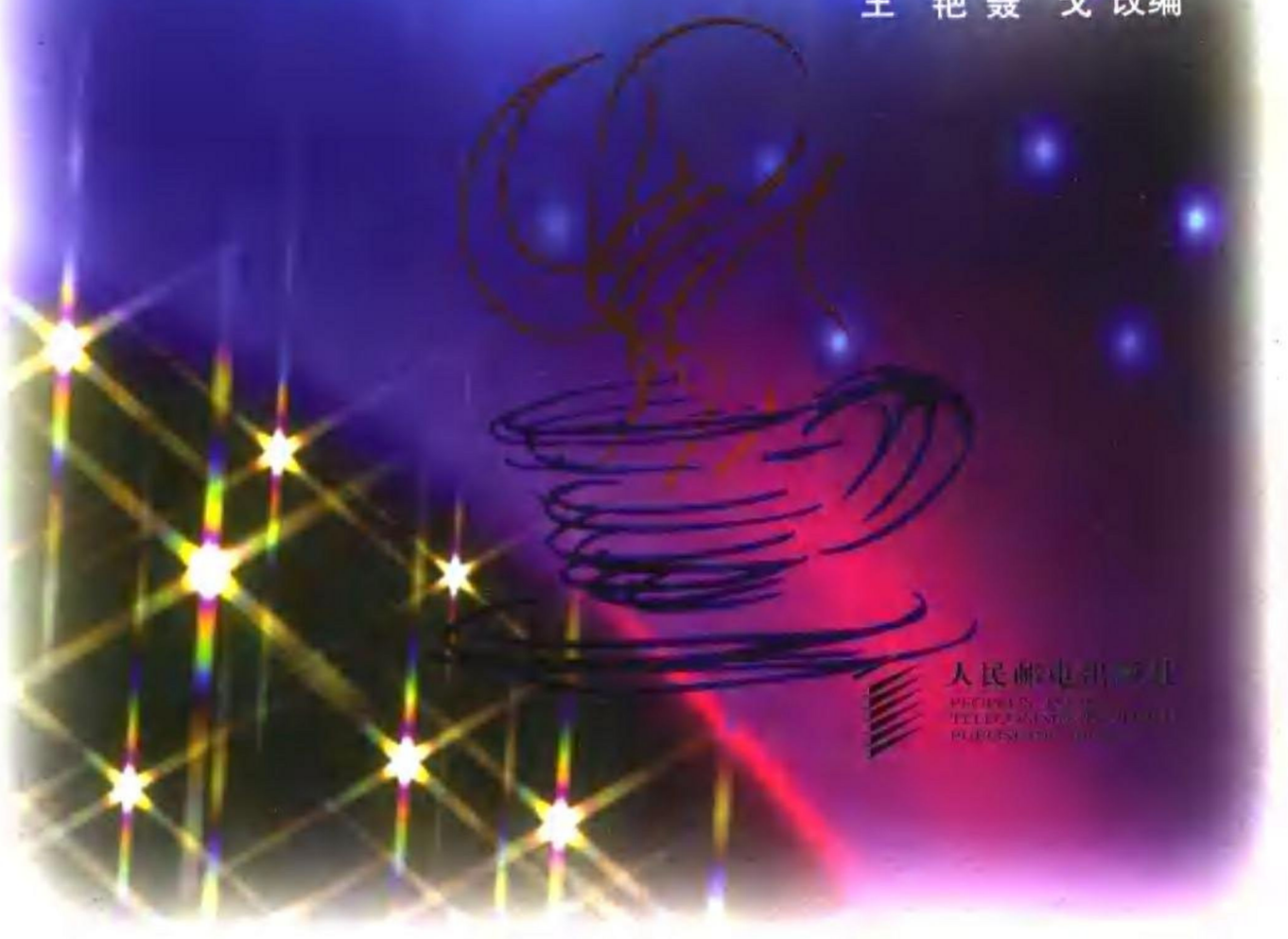
第3波

计算机技术入门提高精通系列丛书

Java & JavaScript 在数据库中的应用

黄国彦 李家铭 张坚琦 编著
王 艳 聂 戈 改编



人民邮电出版社

内 容 简 介

随着Internet的WWW出现而走红的Java语言，是一种简单、面向对象、分布、多线程、与平台无关、安全的解释型计算机编程语言。JavaScript类似于Java，但它不是Java，它是由Netscape公司开发的一种能够嵌入到HTML中的脚本语言。本书通俗地介绍JavaScript在数据库中的应用。全书共分为10章，内容包括：JavaScript的基本知识，JavaScript的变量、数据类型、运算符与表达式，流程控制与语句命令，对象、函数与方法，事件处理器与综合应用等。此外，还分别探讨了JavaScript在Server及未来的网络数据库上的应用。它适合于从事计算机软件开发和网络开发的工程技术人员、软件产品和网络产品经营销售人员、大专院校师生及广大计算机爱好者阅读参考。

计算机技术入门提高精通系列丛书

Java & JavaScript 在数据库中的应用

◆ 编　　著　黄国彦　李家铭　张坚琦
改　　编　王　艳　聂　戈
责任编辑　刘君胜

◆ 人民邮电出版社出版发行　北京崇文区夕照寺街14号
北京顺义向阳胶印刷厂印刷
新华书店总店北京发行所经销

◆开本：787×1092　1/16
印张：11
字数：272千字　　1998年5月第1版
印数：1—5 000册　　1998年5月北京第1次印刷
著作权合同登记　图字：01-97-1665号
ISBN 7-115-07107-1/TP・670

定价：17.00元

出版说明

在计算机技术飞速发展的今天，为了进一步向全社会普及计算机知识，提高计算机应用人员的技术水平，使计算机在各个领域发挥更大作用，也为了促进海峡两岸计算机技术图书的交流，台湾第三波文化事业股份有限公司授权我社陆续组织出版该公司的部分计算机技术书籍。

在组织出版过程中，我们请有关专家在尊重原著的前提下，进行了改编，并对有关图文进行了核对和精心制作。

由于海峡两岸计算机技术名词和术语差异较大，改编者依照有关规定和我们的习惯用法进行了统一整理。

对原书文字叙述中由于海峡两岸不同的语言习惯而造成的差异，我们的处理原则是只要不会造成读者理解上的歧义，一般没做改动，以尊重原著写作风格。另外改编时对原书的一些差错及疏漏之处做了订正。

由于改编和出版时间紧张，本书难免有差错和疏漏，敬请读者指正。

人民邮电出版社

1998.2

版权声明

目 录

第一章 JavaScript基础

JavaScript可以用两种方式嵌入到HTML文件中。一种是被用作函数或语句，另一种是用作事件处理器(event handler)。

首先，我们通过一些最简单的例子来说明JavaScript的用法。

范例1-1：

```
<HTML>
<HEAD>
<TITLE>Example 1-1</TITLE>
<SCRIPT LANGUAGE="JAVASCRIPT">
document.write ("This is a simple example ")
</SCRIPT>
</HEAD>
<BODY>
of JavaScript.
</BODY>
</HTML>
```

此程序的输出结果为：

```
This is a simple example of JavaScript.
```

如图1-1所示。

其中document.write的功能只是将双引号内的字符串输出到屏幕上。这是个非常简单的例子，下一个例子可以说明JavaScript函数的用法。

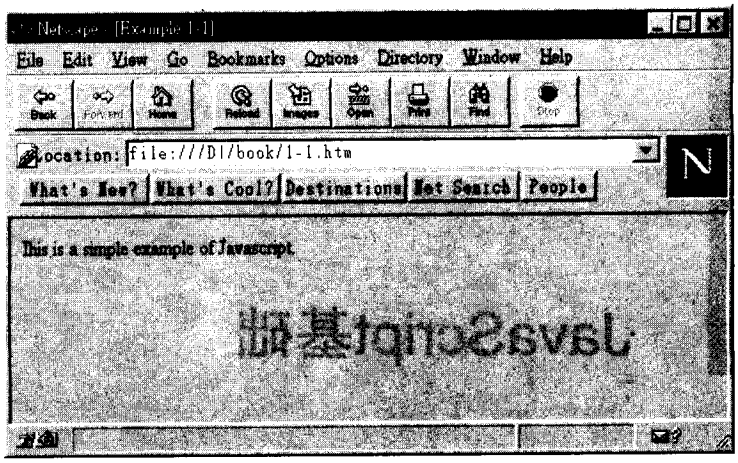


图1-1

范例1-2：

```
<HTML>
<HEAD>
<TITLE>Example 1-2</TITLE>
<SCRIPT LANGUAGE="JAVASCRIPT">
function Add(a,b)
{
    c=a+b
    return c
}
document.write ("a = 1, b = 2, c = a + b = ", Add(1,2))
</SCRIPT>
</HEAD>
<BODY>
<BR>
This show the simple "Add" function.
</BODY>
</HTML>
```

上述例子中定义了一个函数Add，它有两个输入变量，分别为a与b，函数Add的输出结果为c=a+b。当程序执行到document.write中的Add(1,2)时，将会去执行函数Add，于是产生了输出结果“3”，然后由函数document.write输出，如图1-2所示。

接下来的例子将介绍JavaScript的事件处理器，这是一个非常有用也非常特别的功能。事件(event)在程序中指的是某个操作的发生，通常是由于用户按了鼠标的按钮或是移动了鼠标，当然也可以是按了键盘上的按键等，而程序员可以定义事件处理器，当一个事件发生并被检测到时，程序可依此执行一连串的命令。

同样的，事件处理器被嵌入HTML的标记(tag)里。当事件发生时，它所指定的JavaScript程序就会被执行，其标准格式如下：

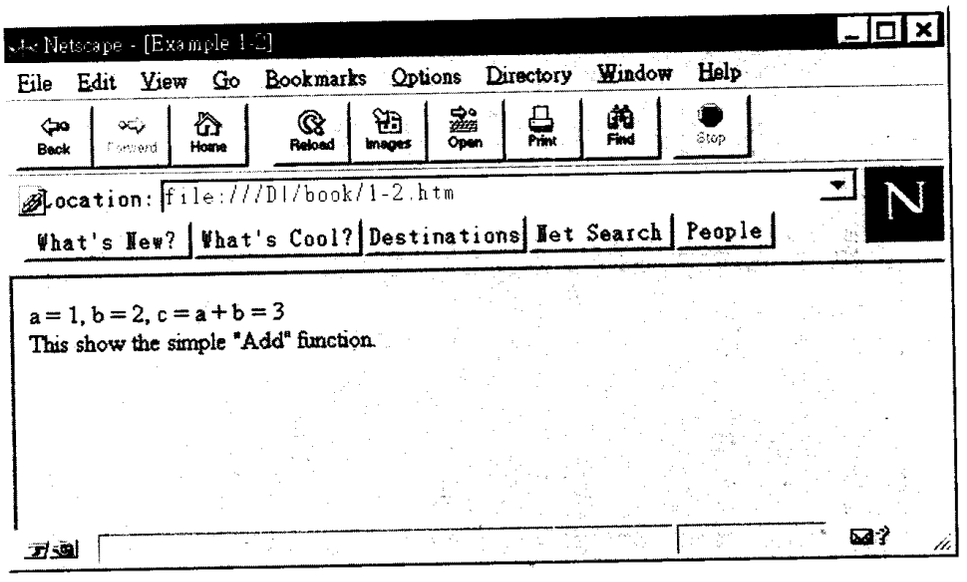


图1-2

<Tag Eventhandler = "指定的JavaScript程序代码或是函数">

其中Tag是指HTML的标记，而Eventhandler则是事件处理器的名称。

以下的两个例子将使读者对JavaScript的事件处理器有初浅的了解，而更高级的部分则会在稍后的章节里做更深入的介绍。

范例1-3：

```
<HTML>
<HEAD>
<TITLE>Example 1-3</TITLE>
<SCRIPT LANGUAGE="JAVASCRIPT">
function demo( )
{
    alert("You just click the button!")
}
</SCRIPT>
</HEAD>
<BODY>
<FORM>
<input type="button" value="Try Me!" onclick="demo( )">
<BR>
</FORM>
</BODY>
</HTML>
```

在此例中，我们首先在表头里定义了一个函数Demo，它的唯一功能是在被调用时打开一个新的警告窗口，内容为“You just click the button!”，而在HTML的<BODY>内我们定义了一个按钮名为“Try Me!”，它是一个事件处理器 (Event handler) 的枢纽，如图1-3所示。

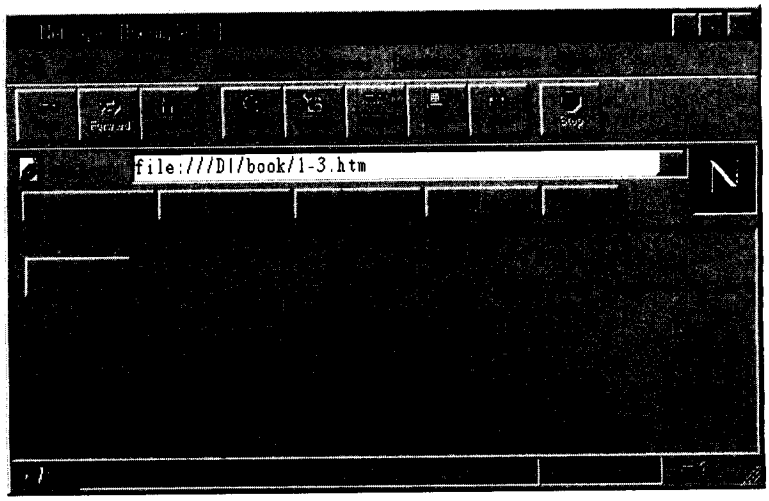


图1-3

当鼠标光标指到“Try Me!”按钮上且按下鼠标左键(缺省值)时，程序就会调用函数Demo，继而在屏幕上显示出警告窗口，如图1-4所示。

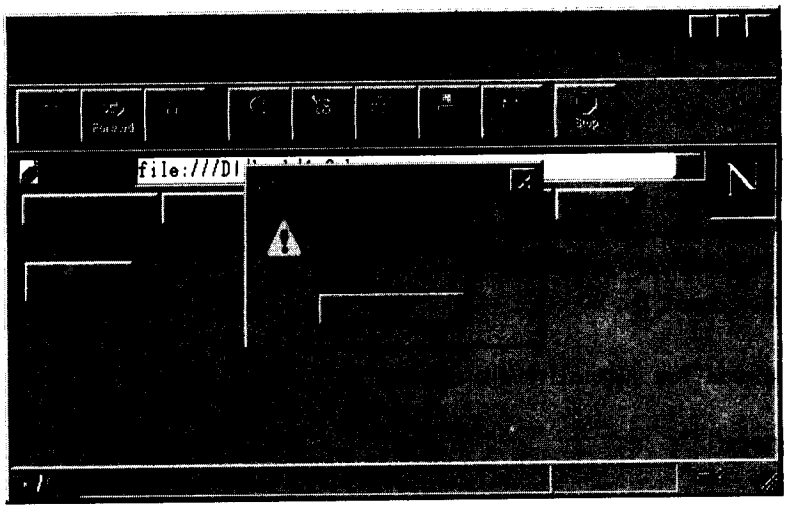


图1-4

接下来的例子对初学者而言也许有一些困难，如果读者对某些关键字的用途不是很清楚的话也没有关系，因为本章目的只是希望让读者对JavaScript的功能有个大概的了解，至于关键字的使用在以后的章节里将会陆续提到。

范例1-4：

```
<HTML>
<HEAD>
<TITLE>Example 1-4</TITLE>
<SCRIPT LANGUAGE="JAVASCRIPT">
```

```
function Price_check(selectForm)
{
 var total_cost = 0
 for (i=0; i < selectForm.Item.length; i++)
     {
      if (selectForm.Item[i].checked==true)
         {
          total_cost=total_cost + eval(selectForm.Item[i].value)
         }
     }
 alert( 'Hello! Your total cost is $ '+ total_cost)
}
</SCRIPT>
</HEAD>
<BODY>
<FORM NAME="MyForm">
<P>
<I>
Mark any item below and Click the button to check the total cost:
</I></B><BR>
<P>
<INPUT TYPE=CHECKBOX NAME="Item" VALUE=1000>CPU - $1000<BR>
<INPUT TYPE=CHECKBOX NAME="Item" VALUE=400>Monitor - $400<BR>
<INPUT TYPE=CHECKBOX NAME="Item" VALUE=300>Printer - $300<BR>
<INPUT TYPE=CHECKBOX NAME="Item" VALUE=99>Modem - $99<BR>
<INPUT TYPE=CHECKBOX NAME="Item" VALUE=50>Speaker - $50<BR>
<P>
<INPUT TYPE="button" VALUE="Calculate the Total Cost" onClick="Price_check(this.form)"
</FORM>
</BODY>
</HTML>
```

在Example1-4中，我们首先在表头里定义了一个函数Price_check，它的功能是在被调用时会将所有选择出的项目的总价加起来，然后利用JavaScript内建警告函数(Alert function)将计算所得的总价另开一个窗口列出。这个程序用到了对象的属性，也用到了循环(For-Next)控制与条件控制(If-then)等，因为在以后的章节里将有更详细的说明，所以我们不打算在这里详谈。另外在程序里还使用到JavaScript的内建函数eval，函数eval的功能是将数据类型是字符串的变量转换为数值类型以便用于数值运算。

当执行程序时，首先出现如图1-5所示的屏幕。

在选择了CPU、Printer以及Modem这三个项目后按下“Calculate the Total Cost”按钮，得到图1-6所示的结果。

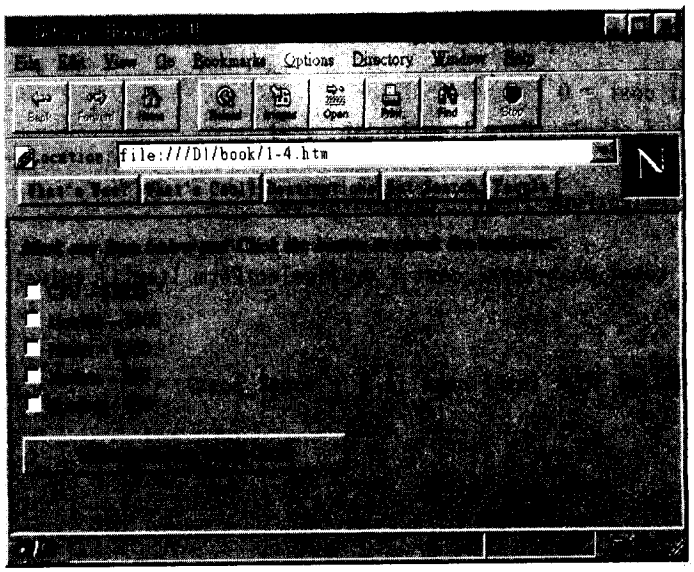


图1-5

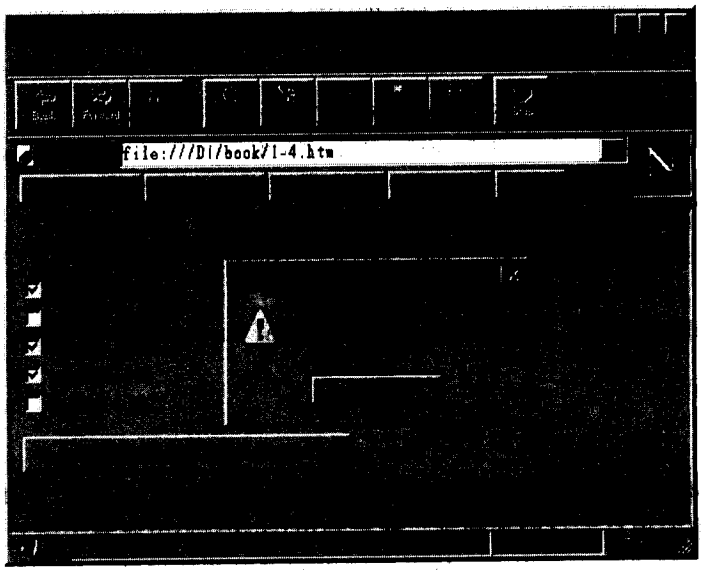


图1-6

通过上面几个简单的例子，相信读者可以感觉到JavaScript与一般的程序设计语言如Basic、FORTRAN、C等类似，所不同的是它可以被嵌入万维网(WWW)的HTML中，使程序员不再局限于以往HTML功能有限的缺点，可以完完全全地将一般程序设计语言的特性带入万维网(WWW)的主页设计中。除此之外也由于JavaScript继承了部分Java语言的特性，像简捷、结构化等优点，使程序员在编写JavaScript时不但能保持一般程序设计语言的特性，而且能加入某些Java语言才有的特性。由于JavaScript可与Java结合使用，所以使程序设计变得更为简单与安全。我们可以将JavaScript简单易写的特性与Java保密及强大功能的特性加以组合。有关Java与JavaScript的合并，在本章的提高篇中还会做进一步的研究。

第二章 JavaScript的变量、数据类型、运算符与表达式

上一章对JavaScript做了一个简单的介绍，从本章开始，我们将对JavaScript进行系统的讲述。

2.1 JavaScript的变量

在高级语言的世界里，主要是使用变量(Variable)来代表计算机的内存地址，也就是存放数据的地方。每一个变量在高级语言里都以英文名字来表示，而不是用数字编码。当我们把数据存入某个变量时，计算机便将所对应的内存地址找出而把此数据存入相对应的内存单元中，如此一来用户所使用的便是十分简单易记的名称而非毫无意义的机器代码了。

同样的，JavaScript 变量名的用法与一般程序语言类似，可以任何字母开头，如A-Z或a-z 或是下划线“_”，但不能用数字0-9做为变量名的第一个字符。

变量名的第二个字符及以后的字符则可以用数字与字母。由于变量名中可以使用下划线“_”，因而我们在取变量名时可以尽量口语化，如“File_Number_One”就比“FileNumberOne”好记多了。其它如Hello_123ABC、_whoareyou、TRYme都是合法的名称。

要注意的一点是JavaScript对大小写字母是敏感的，也就是说“Name”与“name”分别代表不同的变量。

2.2 JavaScript的数据类型

从早期机器语言处理的低级数据类型，如机器码0101111，到现在高级语言如C、PASCAL、FORTRAN、BASIC等，所能处理的整数、浮点数、布尔值、字符、字符串等，数据类型(Data Type)上有很大的变化。程序员不再需要在低级数据类型的处理上大伤脑筋，因为现在数据类型的层次提高了，而JavaScript当然也有与其它高级语言类似的高低级数据类型提供给程序员使用。主要有数字(number)如25、-25、8.367、-0.003，布尔值(Boolean)如“真”(true)或“假”(false)，至于字符串(String)则如“Apple”、“Delicious”等。以下将进行较为详尽的介绍：

1. 整数

JavaScript的整数(Integers)通过3种形式来表示：

(1) 十进制(decimal)：dd....d其中字母d为0-9的整数而且不能用 0开头。一般说来这是最常使用的表示法。

(2) 十六进制(hexadecimal)：0xhh....h其中字母h为0-9、A、B、C、D、E、F的十六进制数。注意须以0X或0x开头才会被视为十六进制的表示符号，此表示法比十进制数用得少，例如：

```
十六进制数的0x3b等于十进制数的59
十六进制数的0X33等于十进制数的51
```

(3) 八进制(octal)：0oo....o其中字母o为0~7的整数。凡是以0做开头的数字则会被视为八进制数来处理，例如：

```
八进制数的013等于十进制数的11
八进制数的025等于十进制数的21
```

2. 浮点数

JavaScript的浮点数(Floating point)可有两种表示法：

(1) dddd.ddd：小数点表示法，小数点左侧为整数部分，右侧为小数部分，例如：

```
1234.567
96343254532
-144.425
```

(2) dddd.dddE+(或-)ddd：科学计数法，把一个数以E(或e)隔开，左侧为真数，右侧为指数(以10为底)，例如：

```
563567.34E12
34343.42332545325E-10
-945.3434e10
```

3. 布尔值

布尔值(Boolean)的表示则很简单，只有两种值true或false。

4. 字符串

双引号或单引号之间的一连串（或零个）字符形成字符串(String)，例如：

```
"dfde"
"dgf454fsg"
"Hello , how are you soing?"
"yes  next line here"
"中文字or English"
```

5. 特殊字符

以下是一些特殊字符(Special Characters)，用来表示某些控制码，在JavaScript中：

- "\b" 表示退格(backspace)。
- "\f " 表示换页(form feed)。
- "\n" 表示换行(new line character)。
- "\r " 表示回车换行(carriage return)。
- "\t " 表示八个空格 (tab character)。

6. 空字符

空字符(Null)是一个特别的值，代表空值(null value)。

7. 退出字符(Escaping Characters)

另一个有用的字符是“\”(back slash)。试试看当您输入如下的程序时，输出结果如何？

范例2-1：

```
<HTML>
<HEAD>
<TITLE>Example 2-1</TITLE>
<SCRIPT LANGUAGE="JAVASCRIPT">
var text1="<B>1: Do you see the quotes beside this word \"hello\"?</B>"
document.write("<CENTER><H2><B>Example2-1</B></H2></CENTER>")
document.write("<HR>")
document.write("<P>")
document.write("Below two lines are the same!")
document.write("<BR>")
document.write("But do not forget to check out the source code")
document.write(" to find out the different methods being used:")
document.write("<P>")
document.write(text1)
document.write("<P>")
document.write( '<B>2: Do you see the quotes beside this word "hello"?</B>' )
</SCRIPT>
```

```
<BODY>
</BODY>
</HTML>
```

输出结果如图2-1所示。

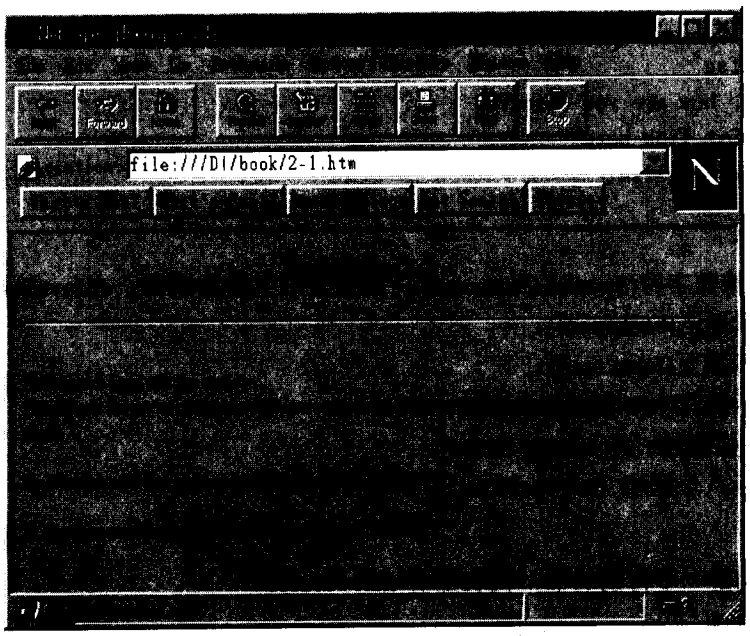


图2-1

另一件值得一提的事是：由于JavaScript是一个松散的语言(这一点不同于Java)，因而并不需要声明它的数据类型，当程序执行时它会自动转换到需要的格式上。这是优点也是缺点，优点是可省下声明每个数值数据的时间；缺点是如果发生了错误，由于数据结构的松散，所以不容易找得到错误的所在。

由于其数据类型自动转换的特性，所以当处理到数值与字符串的表达式时，并不会有太大的困扰，记住一个原则，数值加数值仍然是数值，但数值与字符串相加时，JavaScript则会把数值换成相对应的字符串，然后与其它的字符串连接起来。举例来说：

范例2-2：

```
<HTML>
<HEAD>
<TITLE>Example 2-2</TITLE>
<SCRIPT LANGUAGE="JAVASCRIPT">
var a=100
var b=200
var c=" become a string!"
document.write("<CENTER><H2><B>Example 2-2</B></H2></CENTER>")
document.write("<HR>")

document.write("a = 100 <BR>")
```

```
document.write("b = 200 <BR>")
document.write("c = \" become a string!\" <P>")
document.write("<B>See what is the result of a + b + c ? </B><P>")
document.write("The result is ", a + b + c)
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>
```

其输出结果如图2-2所示。

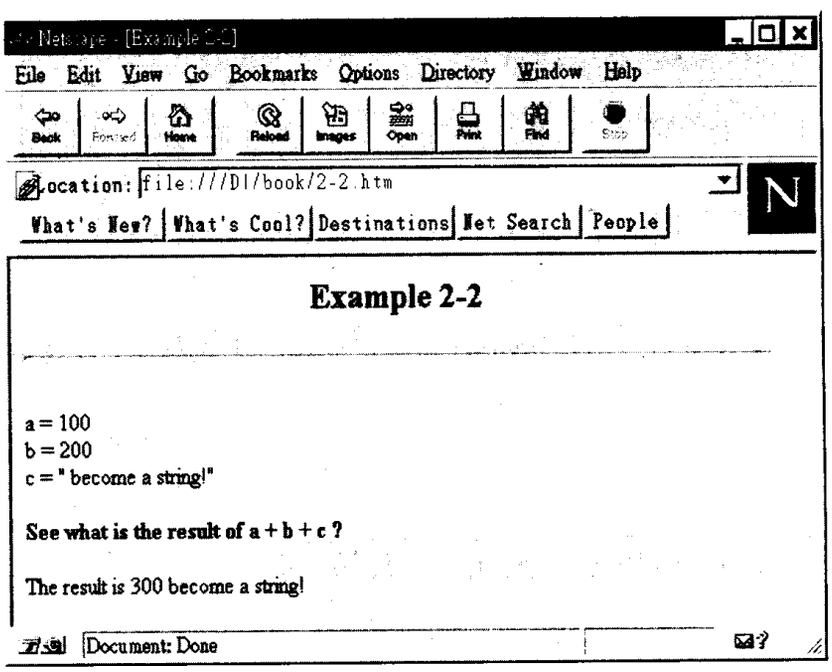


图2-2

有一点要注意的是，当程序处理a + b + c时，由于a、b已先被计算，而得出的结果300再与字符串c=" become a string!"相加时，数值300转换成字符串"300"后与字符串C结合，产生的结果是字符串"300 become a string!"。

如果将程序中第10与第11行：

```
document.write("<B>See what is the result of a + b + c ? </B><P>")
document.write("The result is ", a + b + c)
```

改成：

```
document.write("<B>See what is the result of c + a + b ? </B><P>")
document.write("The result is ", c + a + b)
```

则输出结果如图2-3所示。