第 2 章

JavaScript 的变量与运算符

从本章节可以学习到：

- JavaScript 的变量
- JavaScript 的数据类型
- JavaScript 的运算符
- 数据类型的转换

2-1 JavaScript 的变量

JavaScript 是一种“松散类型的程序设计语言”（Loosely Typed Programming Language），程序变量不需要事先声明就可以直接使用。我们可以将 JavaScript 变量视为一个在程序代码中暂存数据的容器，变量值可以在程序代码中随时使用变量名称读取或者更改变量值。

2-1-1 变量命名与声明

在 JavaScript 变量存储的值可以是三种基本数据类型：“字符串”（String）、“数值”（Number）和“布尔”（Boolean），如下表所示：

基本数据类型	说明
String	内含一个或多个字符，使用“'”或“"”括起的字符串
Number	整数或浮点数
Boolean	true 真和 false 假

在这一节，JavaScript 程序范例的变量只使用了上表三种基本数据类型，关于 JavaScript 详细数据类型的说明，请参阅第 2-2-1 节。

变量的命名

JavaScript 变量名称区分英文字母大小写，counter、Counter 和 COUNTER 是不同的变量，变量名称长度并没有限制，JavaScript 变量的命名原则，如下所示：

- 变量名称不能使用 JavaScript 语法的保留字，即关键词，如下表所示：

break	case	catch	continue	debugger
default	delete	do	else	false
finally	for	function	if	in
instanceof	new	null	return	switch
this	throw	true	try	typeof
var	void	while	with	

- 变量名称的开始字符必须为英文字母的大小写或“_”字符，不能使用数字开头。
- 变量名称除开头字符外，可以是英文字母、数字和“_”符号，不能使用句点“.”，句点是保留给对象使用的运算符。

变量的声明

在 JavaScript 程序代码中使用【var】指令声明变量，如下所示：

```
var strName;
```

上述程序代码声明一个字符串变量 strName，如果需要同时声明多个变量，请使用“,”分隔它们，如下所示：

```
var strName, intBalance;
```

上述程序代码在同一条 var 指令声明了两个变量，一为整数，一为字符串，目前的 JavaScript 程序代码只声明了变量，并没有给变量赋值，所谓变量类型只是使用变量名称的前缀来标记变量所存储数据的类型。

事实上，JavaScript 可以在声明变量的同时给变量赋值，如下所示：

```
var strName = "陈会安";
var intBalance = 1000;
var blnSex = true;
```

上述程序代码声明了 3 个变量并给变量赋了值。

JavaScript 程序：Ch2_1_1.html

在 JavaScript 程序中声明了 3 个变量并给变量赋值了，然后将变量值显示出来，如右图所示：

上述图例显示了 3 个变量值，变量值在声明时就已经赋了值。

程序内容

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04: <meta charset="utf-8"/>
05: <title>Ch2_1_1.html</title>
06: </head>
07: <body>
08: <script>
09: // 变量声明
```

```
10: var strName = "陈会安";
11: var intBalance = 1000;
12: var blnSex = true;
13: // 显示变量的内容
14: document.write("账户名称: " + strName + "<br/>");
15: document.write("性别: " + blnSex + "<br/>");
16: document.write("账户余额: " + intBalance);
17: </script>
18: </body>
19: </html>
```

程序说明

第 10~12 行：声明 3 个变量同时给变量赋值。

第 14~16 行：显示变量值，其中“+”号为字符串连接运算符，可以将变量值和字符串连接起来，进一步说明请参阅第 2-4-1 节数据类型的强制转换和第 2-3-2 节的算术运算符。

2-1-2 赋值语句

在声明变量后如果没有给变量赋值，我们可以使用赋值语句“=”等号来给变量赋值，事实上，在给变量赋值的同时，也指定了变量的数据类型，如下所示：

```
strName = "陈会安";
intBalance = 1000;
```

上述程序代码给变量赋值了，字符串变量 strName 为“陈会安”，整数变量 intBalance 为 1000，不仅如此，我们还可以再次使用赋值语句更改变量的值，如下所示：

```
intBalance = "1000";
```

上述变量 intBalance 的数据类型也随之成为字符串类型，换句话说，JavaScript 变量只是一个暂存数据的容器，变量声明只是声明程序代码需要一个变量的容器，至于变量的数据类型，我们可以使用赋值语句随时更改其数据类型。

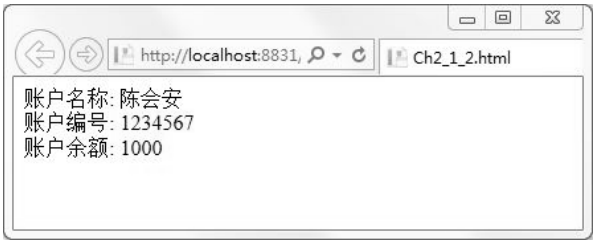
事实上，JavaScript 变量根本就不需要事先声明，如果在程序代码需要使用变量，直接使用赋值语句，即可同时声明并给变量赋值，如下所示：

```
strNo = "1234567";   // 没有声明变量
```

上述程序代码的变量 strNo 没有使用 var 声明，不过，我们依然可以使用赋值语句给变量赋值，同时也创建了这个变量。

JavaScript 程序：Ch2_1_2.html

在 JavaScript 声明变量后，使用赋值语句给变量赋值，然后显示变量的内容，如下图所示：



上述图例显示变量值，这些变量都是使用赋值语句给变量赋的值。

程序内容

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04: <meta charset="utf-8"/>
05: <title>Ch2_1_2.html</title>
06: </head>
07: <body>
08: <script>
09: // 变量声明
10: var strName, intBalance;
11: // 给变量赋值
12: strName = "陈会安";
13: intBalance = 1000;
14: strNo = "1234567";   // 没有声明变量
15: intBalance = "1000"; // 重新指定成字符串
16: // 显示变量的内容
17: document.write("账户名称: " + strName + "<br/>");
18: document.write("账户编号: " + strNo + "<br/>");
19: document.write("账户余额: " + intBalance);
20: </script>
21: </body>
22: </html>
```

程序说明

第 10 行：同时声明 2 个变量 strName 和 intBalance。

第 12~13 行：使用赋值语句给变量赋值。

第 14 行：赋值语句给变量 strNo 赋值并没有先行声明变量。

第 15 行：将原来是整数的变量 intBalance 改为字符串值，换句话说，它的数据类型也改成字符串类型。

第 17~19 行：显示 3 个变量值。

2-1-3　JavaScript 的变量是否存在

JavaScript 程序代码的变量需要使用 var 来声明，或者使用赋值语句来隐藏声明。对于一个变量，程序代码如何知道它是否存在，所谓存在是指变量拥有值，而不是 undefined（未定

义）数据类型，数据类型的详细介绍请参阅第 2-2-1 节。

笔者准备使用第 3 章的 if 条件语句来检查变量是否存在，如下所示：

```
if (intBalance)
   document.write("intBalance 存在<br/>");
else
   document.write("intBalance 不存在<br/>");
```

上述 if 条件检查变量值是否为 undefined，如果 false 表示存在，否则变量不存在。问题是变量 intBalance 需要是已经声明的变量，如果变量根本没有声明，此时的 if 条件语句将导致 JavaScript 运行错误，不过，在浏览器并不会显示错误信息，只是根本忽略此 if 条件。

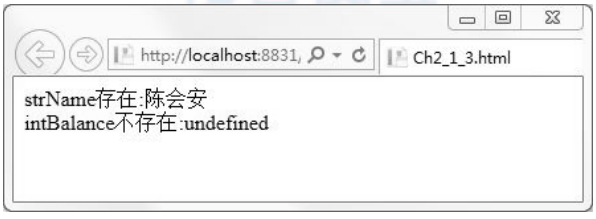
正确的变量检查方法是使用 Window 对象，因为 JavaScript 声明或使用的变量都属于 Window 对象的属性，换句话说，我们只需检查 Window 对象的属性，就可以知道变量是否存在，如下所示：

```
if (window.strName)
   document.write("strName 存在:" + window.strName + "<br/>");
else
   document.write("strName 不存在:" + window.strName + "<br/>");
```

上述 if 条件无论变量是否声明，都可以检查变量是否存在，不过，此 if 条件的程序代码并不适用布尔数据类型的变量检查。

JavaScript 程序：Ch2_1_3.html

在 JavaScript 程序中使用 Window 对象的属性，配合 if 条件检查变量是否存在，笔者分别测试一个有声明的变量和根本没有声明的变量，如下图所示：



上图显示变量 strName 存在，其后为变量值；变量 intBalance 不存在，其变量值为 undefined。

程序内容

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04: <meta charset="utf-8"/>
05: <title>Ch2_1_3.html</title>
06: </head>
07: <body>
```

```
08: <script>
09: // 变量声明
10: var strName = "陈会安";
11: // 检查变量是否存在
12: if (window.strName)
13:   document.write("strName 存在:" + window.strName + "<br/>");
14: else
15:   document.write("strName 不存在:" + window.strName + "<br/>");
16: // 一个不存在的变量
17: if (window.intBalance)
18:   document.write("intBalance 存在:" + window.intBalance + "<br/>");
19: else
20:   document.write("intBalance 不存在:" + window.intBalance + "<br/>");
21: </script>
22: </body>
23: </html>
```

程序说明

第 10 行：声明变量 strName 并赋予了初值，表示变量 strName 存在。

第 12~15 行：if 条件检查变量 strName，如果不是 undefined，就运行第 13 行；否则运行第 15 行。

第 17~20 行：if 条件检查一个不存在的变量 intBalance，因为我们并没有声明此变量。

2-2　JavaScript 的数据类型

JavaScript 变量的基本数据类型除了数值、布尔和字符串类型外，还有对象和数组，特殊数据类型 Null 和 Undefined。

2-2-1　JavaScript 的数据类型

JavaScript 数据类型就是变量存储值的种类，在本节只说明基本和特殊数据类型，对象和数组数据类型请参阅本书后各章节的说明。

数值数据类型（Number Data Type）

JavaScript 数值数据类型的整数和浮点数并没有什么不同，数值数据类型的变量值可以是整数或浮点数，简单地说，数值数据类型就是浮点数据类型。数值数据类型的变量值，如下所示：

- 整数值：整数值包含 0、正整数和负整数，可以使用十进制、八进制和十六进制表示，如果是“0”开头的数值且每个位数的值为 0~7 的整数就是八进制，“0x”开头的数值，位数值为 0~9 和 A~F，这是十六进制，一些整数的范例，如下表所示：

整数值	十进制值	说明
19	19	十进制整数
0182	182	虽然是 0 开头，不过因为有 8，所以不是八进制 而是十进制整数
0377	255	八进制整数
0xff	255	十六进制整数
0x3e7	999	十六进制整数

- 浮点数：浮点数是整数加上小数，其范围最大为±1.7976931348623157xE^{308}，最小为 ±5xE^{-324}，使用“e”或“E”符号代表 10 为底的指数，一些浮点数的范例，如下表所示：

浮点数	十进制值	说明
0.0005	0.0005	浮点数
.0005	0.0005	浮点数
5e-4	0.0005	使用 e 指数的浮点数

JavaScript 数值数据类型拥有一些特殊值的字符串，通常是出现在数值数据类型发生错误时，如下表所示：

数值的特殊值字符串	说明
NaN	Not a number，当算术表达式的运算结果是不正确数据时， 例如：字符串或 Undefined
Positive Infinity	数值太大超过 JavaScript 正数值的范围
Negative Infinity	数值太大超过 JavaScript 负数值的范围
Positive and Negative 0	JavaScript 用来区分+0 和-0

字符串数据类型（String Data Type）

字符串可以包含 0 或多个 Unicode 字符，包含文字、数值和标点符号，字符串数据类型是用来存储文字内容的变量，JavaScript 程序代码的字符串需要使用“""”或“'”符号括起。一些中英文的字符串范例，如下所示：

字符串变量值	显示的字符串
"JavaScript"	JavaScript
"1234567"	1234567
'陈会安'	陈会安
'"陈会安"是本书的作者'	"陈会安"是本书的作者
"I'm an author."	I'm an author.

备注

“统一码”（Unicode）是由 Unicode Consortium 组织所制定的一个能包括全世界文字的内码集，包含 GB2312 和 Big5 的所有内码集，即 ISO 10646 内码集。拥有常用的两种编码方式：UTF-8 为 8 位编码；UTF-16 为 16 位的编码。

JavaScript 不支持单一字符的函数，例如：Visual Basic 或 C/C++语言的 chr()函数，我们只能使用单一字符的字符串，例如："J"、'c'，如果连一个字符都没有，""就是空字符串。

- **布尔数据类型（Boolean Data Type）**

布尔数据类型只有两个值 true 和 false，主要是用在第 3 章条件和循环控制的条件判断，以便决定继续运行哪一个程序块的程序代码，或是判断循环是否结束。

- **Null 数据类型**

Null 数据类型只有一个值 null，null 是一个关键词并不是 0，如果变量值为 null，表示变量没有值或不是一个对象。

- **Undefined 数据类型**

Undefined 数据类型指的是一个变量有声明，但是不曾给变量赋值，或是一个对象属性根本不存在。

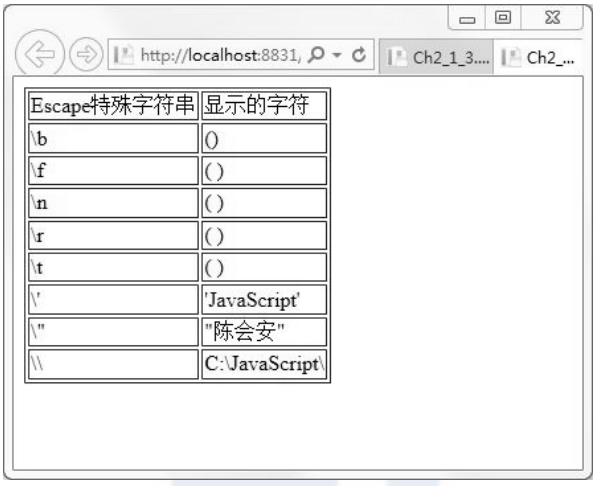
2-2-2　Escape 转义字符

JavaScript 提供 Escape 转义字符，它们是使用“\”符号开头的字符，可以在字符串数据类型的变量值中显示无法使用键盘输入的特殊字符，如下表所示：

Escape 转义字符	说明
\b	Backspace，Backspace 键
\f	Form feed
\n	LF，Line feed 换行符号
\r	CR，Enter 键
\t	Tab 键
\'	“'”符号
\"	“"”符号
\\	“\”符号

JavaScript 程序：Ch2_2_2.html

在 JavaScript 程序中显示 Escape 转义字符对应的字符，不过，大部分字符无法在浏览器正确地显示，如下图所示：



Escape特殊字符串	显示的字符
\b	()
\f	( )
\n	( )
\r	( )
\t	( )
\'	'JavaScript'
\"	"陈会安"
\\	C:\JavaScript\

上述表格显示 Escape 转义字符，只有最后 3 个字符可以正确显示，其他字符通常用于文本文件处理时对输出文字内容的控制，例如：“\n”为换行。

程序内容

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04: <meta charset="utf-8"/>
05: <title>Ch2_2_2.html</title>
06: </head>
07: <body>
08: <script>
09: document.write("<table border='1'>");
10: document.write("<tr><td>Escape 特殊字符串</td>");
11: document.write("<td>显示的字符</td></tr>");
12: document.write("<tr><td>\\b</td><td>(\b)</td></tr>");
13: document.write("<tr><td>\\f</td><td>(\f)</td></tr>");
14: document.write("<tr><td>\\n</td><td>(\n)</td></tr>");
15: document.write("<tr><td>\\r</td><td>(\r)</td></tr>");
16: document.write("<tr><td>\\t</td><td>(\t)</td></tr>");
17: document.write("<tr><td>\\'</td><td>\'JavaScript\'</td></tr>");
18: document.write("<tr><td>\\\"</td><td>\"陈会安\"</td></tr>");
19: document.write("<tr><td>\\\\</td><td>C:\\JavaScript\\</td></tr>");
20: document.write("</table>");
21: </script>
22: </body>
23: </html>
```

程序说明

第 9~20 行：输出 HTML 表格标签。

第 12~19 行：显示 Escape 转义字符。

2-3　JavaScript 的运算符

JavaScript 语句的表达式都是由运算符和操作数组成，JavaScript 拥有完整的算术、赋值、位和逻辑运算符，一些表达式的范例，如下所示：

```
a + b - 1
a >= b
a > b && a > 1
```

上述表达式变量 a、b 和数值 1 是操作数；“+”、“-”、“>=”、“>”和“&&”是运算符。

2-3-1　运算符的优先级

因为 JavaScript 提供多种运算符，而且在同一个表达式允许使用多种运算符，为了让表达式能够得到相同的运算结果，表达式会以运算符默认的优先级进行运算，也就是我们所熟知的“先乘除后加减”，如下所示：

```
a + b * 2
```

上述表达式先计算 b*2 后才和 a 相加，这就是运算符的优先级。JavaScript 运算符的优先级（从高到低排列），如下表所示：

运算符	说明
()	括号
!、-、++、--	逻辑运算符 NOT、算术运算符的负号、递增和递减
*、/、%	算术运算符的乘、除法和求余数
+、-	算术运算符的加法和减法
<<、>>、>>>	位运算符左移、右移和无符号右移
>、>=、<、<=	比较运算符大于、大于等于、小于和小于等于
==、!=	比较运算符等于和不等于
&	位运算符 AND
^	位运算符 XOR
|	位运算符 OR

（续表）

运算符	说明
&&	逻辑运算符 AND
||	逻辑运算符 OR
?:	条件运算符
=、op=	赋值运算符

在上表的条件运算符 ?: 可以在表达式建立条件控制来给变量赋不同的值，如同 if 条件语句，详细说明请参阅第 3 章。

2-3-2 算术运算符

JavaScript 算术运算符拥有常用的数学运算符，大部分操作数是数值，不过，加号可以连接两个字符串变量。各种算术运算符的说明与范例（变量 a 的值为 10），如下表所示：

运算符	说明	表达式范例
-	负号	-7
++	递增运算	a++ = 11
--	递减运算	a-- = 9
*	乘法	5 * 6 = 30
/	除法	7 / 2 = 3.5
%	求余数	7 % 2 = 1
+	加法或字符串连接	4 + 3 = 7
-	减法	4 – 3 = 1

上表递增和递减运算++和--可以置于变量之前或之后，如下所示：

```
x++;
--yy;
```

上述算术运算符在变量之前，变量值会立刻改变；之后则是在运行表达式后才改变，如下所示：

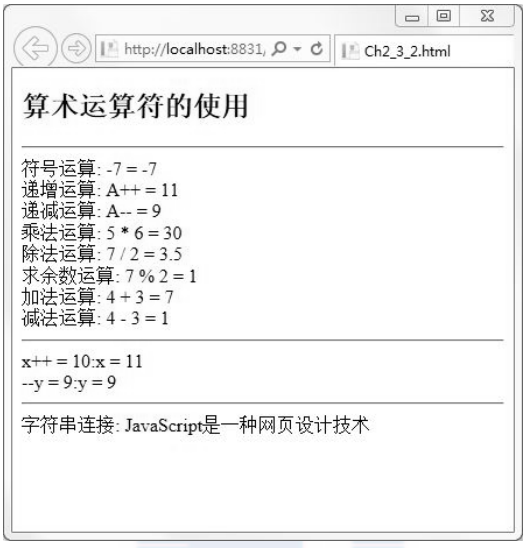
```
x = 10;
y = 10;
document.write("x++ = " +x+++":x = " + x + "<br/>");
document.write("--y = " +--y+":y = " + y + "<br/>");
```

上述变量 x 和 y 的初始值为 10，后面两行程序代码的第一行是 x++，运算符在后所以之后才改变，第一个 x++值仍然为 10；第二个 x 为 11，最后一行的--y 运算符在前，所以第一

个为 9；第二个也是 9。

JavaScript 程序：Ch2_3_2.html

在 JavaScript 程序中测试上表的各种 JavaScript 算术运算符，如下图所示：



上述图例在第二条水平线前为算术运算符，中间测试递增和递减运算符，其位置分别在变量之前和之后，最后使用“+”加号连接 2 个字符串变量。

程序内容

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04: <meta charset="utf-8"/>
05: <title>Ch2_3_2.html</title>
06: </head>
07: <body>
08: <h2>算术运算符的使用</h2>
09: <hr/>
10: <script>
11: var x, y;
12: var intInc = 10;
13: var intDec = 10;
14: var strTitle1 = "JavaScript 是"
15: var strTitle2 = "一种网页设计技术"
16: document.write("负号运算: -7    = " + -7 + "<br/>");
17: intInc++;
18: document.write("递增运算: A++ = " + intInc + "<br/>");
19: intDec--;
20: document.write("递减运算: A-- = " + intDec + "<br/>");
21: document.write("乘法运算: 5 * 6 = " + 5*6 + "<br/>");
```

```
22: document.write("除法运算: 7 / 2 = " + 7/2 + "<br/>");
23: document.write("求余数运算: 7 % 2 = " + 7%2 + "<br/>");
24: document.write("加法运算: 4 + 3 = " + (4+3) + "<br/>");
25: document.write("减法运算: 4 - 3 = " + (4-3) + "<br/><hr/>");
26: // 测试 Pre-/Post- 运算符
27: x = 10;
28: y = 10;
29: document.write("x++ = " +x+++":x = " + x + "<br/>");
30: document.write("--y = " +--y+":y = " + y + "<br/><hr/>");
31: document.write("字符串连接: " + (strTitle1+strTitle2) + "<br/>");
32: </script>
33: </body>
34: </html>
```

程序说明

第 11~15 行：声明变量和赋初值。

第 16~25 行：测试算术运算符。

第 27~30 行：测试 x++和--y 表达式。

第 31 行：连接 2 个字符串变量。

2-3-3 逻辑与比较运算符

逻辑与比较运算符主要用于第 3 章将介绍的循环和条件语句的判断条件，true 为真；false 为假。比较运算符的说明和范例，如下表所示：

运算符	说明	表达式范例	运算结果
==	等于	6= 3	false
!=	不等于	6!= 3	true
<	小于	6< 3	false
>	大于	6> 3	true
<=	小于等于	6<= 3	false
>=	大于等于	6 >=3	true

如果条件不止一个，就需要使用逻辑运算符连接各比较表达式，其说明如下表所示：

运算符	说明
!	NOT 非，返回操作数相反的值，true 成 false，false 成 true
&&	AND 与，连接的两个操作数都为 true，表达式结果为 true
||	OR 或，连接的两个操作数，任何一个为 ture，表达式结果为 true

JavaScript 程序：Ch2_3_3.html

在 JavaScript 程序中测试上表的逻辑和比较运算符，如下图所示：

上述图例显示各种逻辑和比较表达式的运行结果，最后两行是逻辑运算符连接两个比较表达式。

程序内容

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04: <meta charset="utf-8"/>
05: <title>Ch2_3_3.html</title>
06: </head>
07: <body>
08: <h2>逻辑与比较运算符的使用</h2>
09: <hr/>
10: <script>
11: var blnA = true;
12: var blnB = false;
13: document.write("NOT 运算: !A 结果为 "+ (!blnA)+"<br/>");
14: document.write("小于: 6<3  结果为 "+(6<3)+"<br/>");
15: document.write("大于: 6>3  结果为 "+(6>3)+"<br/>");
16: document.write("小于等于: 6<=3 结果为 "+(6<=3)+"<br/>");
17: document.write("大于等于: 6>=3  结果为 "+(6>=3)+"<br/>");
18: document.write("等于: 6==3  结果为 "+ (6==3)+"<br/>");
19: document.write("不等于: 6!=3  结果为 "+(6!=3)+"<br/>");
20: document.write("AND 运算: A && B 结果为 "+ (blnA && blnB)+"<br/>");
21: document.write("OR 运算: A || B 结果为 "+ (blnA || blnB)+"<br/>");
22: </script>
23: </body>
24: </html>
```

程序说明

第 11~12 行：声明 2 个变量并赋初值。

第 13 行：NOT 运算。

第 14~19 行：测试比较运算符。

第 20~21 行：测试 AND 和 OR 逻辑运算符。

2-3-4 位运算符

JavaScript 支持位运算符，能够进行二进制值的位运算，我们可以向左移或右移几个位，或运行 NOT、AND、XOR 和 OR 的位运算。位运算符的说明与范例，如下表所示：

运算符	A	B	C	D	范例	结果	说明
~	1(01)				~A	-2(10)	NOT 运算
<<			3(11)		C<< 2	12(1100)	左移运算
>>		2(10)			B >> 1	1(1)	右移运算
>>>				16(1000)	D >>> 1	8(0100)	无符号右移
&	1(01)		3(11)		A & C	1(01)	AND 运算
^	1(01)	2(10)			A ^ B	3(11)	XOR 运算
|	1(01)	2(10)			A | B	3(11)	运算

上表变量值为十进制，括号中的数值是二进制值，无符号右移运算>>>是在右移几个位后，在左边填入 0。

NOT、AND 和 OR 运算的说明请参阅上一节的逻辑运算符。XOR 运算连接的二进制值中，各位只需任意一个为 1，其结果则为 1，如果同为 0 或 1 时结果则为 0。位运算符的结果，a 和 b 各代表一个位，其真假值表如下表所示：

a	b	NOT a	NOT b	a AND b	a OR b	a XOR b
1	1	0	0	1	1	0
1	0	0	1	0	1	1
0	1	1	0	0	1	1
0	0	1	1	0	0	0

JavaScript 程序：Ch2_3_4.html

在 JavaScript 程序中测试上表说明的各种位运算符，如下图所示：

程序内容

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04: <meta charset="utf-8"/>
05: <title>Ch2_3_4.html</title>
06: </head>
07: <body>
08: <h2>位运算符的使用</h2>
09: <hr>
10: <script>
11: var intA = 1;  // 0001
12: var intB = 2;  // 0010
13: var intC = 3;  // 0011
14: var intD = 16; // 1000
15: document.write("NOT 运算: ~A = " + (~intA) + "<br/>");
16: document.write("左移运算: C << 2 = " + (intC<<2) + "<br/>");
17: document.write("右移运算: B >> 1 = " + (intB>>1) + "<br/>");
18: document.write("无符号右移运算: D >>> 1 = " + (intD>>>1) + "<br/>");
19: document.write("AND 运算: A & C = " + (intA & intC) + "<br/>");
20: document.write("XOR 运算: A ^ B = " + (intA ^ intB) + "<br/>");
21: document.write("OR 运算: A | B = " + (intA | intB) + "<br/>");
22: </script>
23: </body>
24: </html>
```

程序说明

第 11~14 行：声明测试变量和赋初值，注释中的数值为该变量的二进制值。

第 15~21 行：使用声明的变量测试各种位运算，并且显示结果。

2-3-5　赋值运算符

JavaScript 赋值运算符除了赋值语句“=”外，赋值运算符能够配合其他运算符来简化表

达式，建立出简洁的算术或位表达式，如下表所示：

运算符	范例	相当于表达式	说明
=	x = y	N/A	赋值语句
+=	x+ = y	x = x + y	数值相加或字符串连接
-=	x -= y	x = x - y	减法
*=	x *= y	x = x * y	乘法
/=	x /= y	x = x / y	除法
%=	x %= y	x = x % y	求余数
<<=	x <<= y	x = x << y	位左移 y 位
>>=	x >>= y	x = x >> y	位右移 y 位
>>>=	x >>>= y	x = x >>> y	无符号右移 y 位
&=	x &= y	x = x & y	位 AND 运算
|=	x |= y	x = x | y	位 OR 运算
^=	x ^= y	x = x ^ y	位 XOR 运算

2-4 数据类型的转换

“数据类型转换”（Type Conversions）是因为同一个表达式可能有多个不同数据类型的变量或值。例如：在表达式拥有整数和浮点数的变量或值时，就需要进行类型转换。

2-4-1 数据类型的强制转换

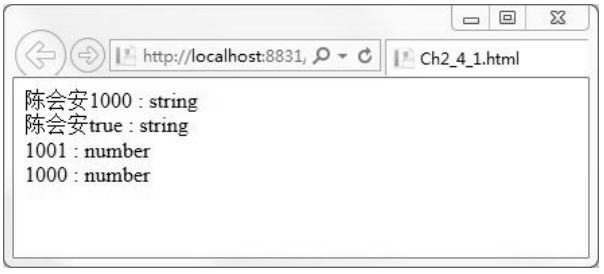
因为 JavaScript 是一种松散类型的程序设计语言，所谓变量的数据类型是指变量值的数据类型。JavaScript 表达式一般情况下都会强制进行类型转换，因为 JavaScript 表达式的操作数需要相同的类型，数据类型的强制转换方式如下表所示：

表达式	类型强制转换的处理
数值和字符串相加	数值会强制转换成字符串
布尔和字符串相加	布尔会强制转换成字符串
布尔和数值相加	布尔会强制转换成数值

布尔值 true 在强迫转换成字符串时为"true"，数值为 1，值 false 转换成字符串"false"，数值为 0。

JavaScript 程序：Ch2_4_1.html

在 JavaScript 程序中测试上表 3 种数据类型的强制转换，使用的是加法的算术运算符，运算符说明详见第 2-3-2 节，在转换后使用 document.write()方法显示转换结果和数据类型，其中使用了下一节的 typeof()函数，如下图所示：



上述图例显示 3 种数据类型的强迫转换结果，其数据类型分别为字符串、字符串、数值和数值，最后两行为布尔加数值，分别为 true 和 false，可以看出转换成数值 1 和 0。

程序内容

```
01: <!DOCTYPE html>
02: <html>
03: <head>
04: <meta charset="utf-8"/>
05: <title>Ch2_4_1.html</title>
06: </head>
07: <body>
08: <script>
09: // 变量声明
10: var strName, intBalance, blnSex;
11: // 给变量赋值
12: strName = "陈会安";
13: intBalance = 1000;
14: blnSex = true;
15: // 显示数据类型的强制转换的结果
16: output = strName + intBalance; // 字符串加数值
17: document.write(output + " : " + typeof(output) + "<br/>");
18: output = strName + blnSex;  // 字符串加布尔
19: document.write(output + " : " + typeof(output) + "<br/>");
20: output = blnSex + intBalance; // 布尔加数值
21: document.write(output + " : " + typeof(output) + "<br/>");
22: blnSex = false;  // 重设布尔值
23: output = blnSex + intBalance; // 布尔加数值
24: document.write(output + " : " + typeof(output) + "<br/>");
25: </script>
26: </body>
27: </html>
```

程序说明

第 10 行：同时声明 3 个变量。

第 12~14 行：分别给变量赋值为字符串、数值和布尔。

第 16~17 行：字符串和数值的强制转换，在第 16 行显示结果和数据类型，它使用 typeof() 函数获取数据类型。

第 18~19 行：字符串和布尔的强制转换。

第 20~24 行：布尔加数值的转换，第 1 个布尔值为 true，第 23~24 行也是布尔加数值，不过此时的布尔值为 false。

2-4-2 数据类型的转换函数

JavaScript 虽然在进行运算时会自动进行数据类型的强迫转换，不过，JavaScript 仍然提供了数个函数和运算符来进行数据类型的转换。

parseInt()函数

将字符串变量值开头的数值转换成整数，如果字符串没有数值，就返回 NaN(Not a number)，在转换时可以指定十六、十和八进制。一些转换的范例，如下表所示：

parseInt()函数	值	说明
parseInt("3 page")	3	字符串开头为数值
parseInt("3.2")	3	虽然是浮点值的字符串，不过只取出整数部分
parseInt("Page 3")	NaN	字符串开头不是数值
parseInt("18ff 值", 16)	6399	将字符串转换成 16 进制数值，也就是 18ff
parseInt("18ff 值", 10)	18	将字符串转换成 10 进制数值，也就是 18
parseInt("18ff 值", 8)	1	将字符串转换成 8 进制数值，因为数值不能超过 8，所以为 1

parseFloat()函数

将字符串变量值开头的浮点数转换成浮点数，如果字符串没有数值，就返回 NaN(Not a number)，如下表所示：

parseFloat()函数	值	说明
parseFloat("3.2")	3.2	字符串开头为浮点数
parseFloat("Page 3.2")	NaN	字符串开头不是浮点数

eval()函数

将表达式的字符串参数当作表达式，函数可以返回表达式的计算结果，一些范例和说明

如下表所示：

eval()函数	值	说明
eval("20 + 4 * 5")	40	算术表达式
eval("intBalance = 1000")	1000	赋值语句
eval("5 > 4")	true	逻辑或比较表达式

typeof()函数

typeof()函数可以获得变量的数据类型，也就是 string、number、boolean、undefined 和 object 等数据类型。

备注

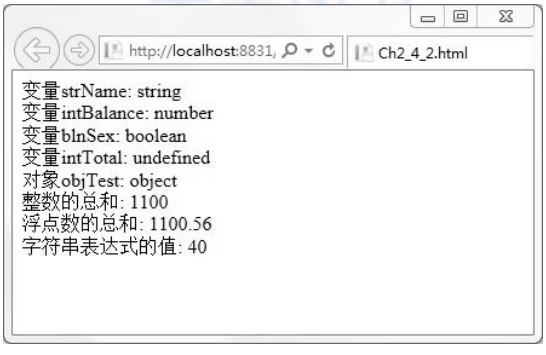
JavaScript 变量值如果为 null，我们可以使用 typeof()函数检查变量类型为 object，而不是 null 数据类型，如下所示：

```
<script>
var strName = null;
document.write(typeof(strName) + "<br/>");
</script>
```

上述程序代码 typeof(strName)的值为 object。

JavaScript 程序：Ch2_4_2.html

在 JavaScript 程序中使用数据转换函数进行变量数据的转换，typeof()函数检查变量的数据类型，如下图所示：



上图运行结果的前 5 行显示 typeof()函数获取的变量数据类型，最后 3 行分别使用 parseInt()、parseFloat()和 eval()函数取得整数、浮点数和表达式的计算结果。

程序内容

```
01: <!DOCTYPE html>
```

```
02: <html>
03: <head>
04: <meta charset="utf-8"/>
05: <title>Ch2_4_2.html</title>
06: </head>
07: <body>
08: <script>
09: // 变量声明
10: var strName, intBalance, blnSex;
11: var strNo, objTest, intTotal, strNo;
12: // 给变量赋值
13: strNo = "100.56 的编号"
14: strName = "陈会安";
15: intBalance = 1000;
16: blnSex = true;
17: objTest = new Object();
18: // 使用 typeof() 显示变量的数据类型
19: document.write("变量 strName: " + typeof(strName) +"<br/>");
20: document.write("变量 intBalance: " + typeof(intBalance) +"<br/>");
21: document.write("变量 blnSex: " + typeof(blnSex) +"<br/>");
22: document.write("变量 intTotal: " + typeof(intTotal) +"<br/>");
23: document.write("对象 objTest: " + typeof(objTest) +"<br/>");
24: // 使用 parseInt()
25: intTotal = parseInt(strNo) + intBalance;
26: document.write("整数的总和: " + intTotal + "<br/>");
27: // 使用 parseFloat()
28: intTotal = parseFloat(strNo) + intBalance;
29: document.write("浮点数的总和: " + intTotal + "<br/>");
30: // 使用 eval()
31: intTotal = eval("20 + 4 * 5");
32: document.write("字符串表达式的值: " + intTotal + "<br/>");
33: </script>
34: </body>
35: </html>
```

程序说明

第 10~11 行：声明 7 个变量。

第 13~16 行：给变量赋值。

第 17 行：使用 new 运算符创建对象。

第 19~23 行：显示各变量的数据类型。

第 25~31 行：分别测试使用 parseInt()、parseFloat()和 eval()函数。

（　）1. 请问下列哪一个不是 JavaScript 合法的变量名称？

A. numOfcars　　B. num_of_cars

C. number　　D. num.of.cars

（　）2. 请问 JavaScript 变量使用下列哪一个关键词进行声明？

A. var　　B. dim　　C. def　　D. type

（　）3. 请问下列哪一种 JavaScript 数据类型变量只有两种值 true 和 false？

A. 布尔　　B. 数值　　C. Null　　D. 字符串

（　）4. 请问数学表达式：6+3*4-3 的运算结果是什么？

A. 11　　B. 13　　C. 15　　D. 17

（　）5. 请问下列哪一个 JavaScript 运算符的优先级最高？

A. “+”　　B. “*”　　C. “/”　　D. “++”

6. 请问如何检查 JavaScript 的变量是否存在？

7. 请编写一个 JavaScript 程序，其中使用类型转换函数，将字符串转换成十、八和十六进制值。

8. 请编写一个 JavaScript 程序，其中使用表格标签页面显示 NOT、AND、OR 和 XOR 位运算的各种结果，操作数分别为 1 和 0。