17.9.4 引用类型转换

上表介绍了把对象转换为原始值的基本方法，但是如何转换，以及转换的结果都没有说明，下面结合几个很容易忽略的问题进行详细解释。

1．对象在逻辑运算环境中的转换

如果把非空对象用在逻辑运算环境中，则对象被转换为 true。这包括所有类型的对象，即使是值为 false 的包装对象也为 true。

【示例 1】下面代码创建 3 个不同类型的对象，然后在逻辑与运算中，可以看到它们全部为 true。

```
var a = new Boolean(false);
var b = new Number(0);
var c = new String("");
a && alert(a);   // a 转换为布尔值为 true，但是提示它的字符串转换值为"false"
b && alert(b);   // b 转换为布尔值为 true，但是提示它的字符串转换值为"0"
c && alert(c);   // c 转换为布尔值为 true，但是提示它的字符串转换值为""
```

2．对象在数值运算环境中的转换

如果对象用在数值运算环境中，则对象会被自动转换为数字，如果转换失败，则返回值 NaN。具体转换过程如下。

首先，调用对象的 valueOf()方法，返回对象自身的值。大多数对象都继承了 Object 对象的 valueOf()。valueOf()方法仅能够返回自身值，但是不会转换值的类型，所以 valueOf()方法取出的值并非都是数值。

如果对象自身值不为数值，就会调用对象的 toString()方法，把对象自身值转换为字符串。然后调用 parseInt()或 parseFloat()函数把字符串转换成数字。

如果上述方法不能够成功，JavaScript 会尝试通过强制方法把对象转换为数值。如果成功则已，不成功就返回 NaN。

【示例 2】下面代码使用 Boolean()构造器把布尔值 true 转换为布尔型对象，然后再通过 a - 0 数值运算，把布尔型对象转换为数字 1。

```
var a = new Boolean(true);        // 把 true 封装为对象
alert(a.valueOf());               // 测试该对象的值为 true
alert(typeof (a.valueOf()));      // 测试值的类型为 boolean
a = a - 0;                        // 投放到数值运算环境中
alert(a);                         // 返回值为 1
alert(typeof a);                  // 再次测试它的类型，则为 number
```

JavaScript 自动转换对象 a 到数字的过程如下。

首先，直接使用 valueOf()方法取值，没有成功。

```
a = a.valueOf();                  // 取出对象自身值
alert(a);                         // 返回 true
alert(typeof a);                  // 返回类型为 boolean
```

然后，把对象自身值转换为字符串，再次尝试转换，没有成功。

```
a = a.valueOf();                  // 取出对象自身值
a = a.toString();                 // 转换为字符串，返回"true"
a = parseFloat(a);                // 转换为数值
alert(a);                         // 返回为 NaN
alert(typeof a);                  // 返回类型为 number
```

最后，尝试强制转换，则成功。

```
a = a.valueOf();                  // 取出对象自身值
a = Number(a);                    // 强制转换
alert(a);                         // 返回 1
alert(typeof a);                  // 返回类型为 number
```

3．数组在数值运算环境中的转换

当数组被用在数值运算环境中时，数组将根据包含的元素来决定转换的值。

- 如果为空数组，则被转换为数值 0。当数组为空时，JavaScript 将调用 toString()方法把数组转换为空字符串，然后再将空字符串强制转换为数值 0。
- 如果数组仅包含一个数字元素，则被转换为该数字的数值。例如：

```
var a = [5];
a = a * 1;                    // 投放到数值运算环境中
alert(a);                     // 返回数值 5
alert(typeof a);              // 返回类型为 number
```

- 如果数组包含多个元素，或者仅包含一个非数字元素，则返回 NaN。例如：

```
var a = [true];
a = a * 1;
alert(a);                     // 返回数值 NaN
alert(typeof a);              // 返回类型为 number
```

4．对象在模糊运算环境中的转换

当对象用于字符串环境中时，JavaScript 能够调用 toString()方法把对象转换为字符串再进行相关计算。而在数值运算环境中时，则会根据上面两小节介绍的方法进行转换操作。但是，在 JavaScript 中有两处运算环境比较模糊：加号运算符和比较运算符。当值进行加号运算或者比较运算时，即可以作用于数值，也可以作用于字符串。

- 当对象与数值进行加号运算时，则会尝试把对象转换为数值，然后参与求和运算。如果不能够转换为有效数值，则执行字符串连接操作。例如：

```
var a = new String("a");      // 字符串封装为对象
var b = new Boolean(true);    // 布尔值封装为对象
a = a + 0;                    // 加号运算
b = b + 0;                    // 加号运算
alert(a);                     // 返回字符串"a0"
alert(b);                     // 返回数值 1
```

- 当对象与字符串进行加号运算时，则直接转换为字符串，进行连接操作。例如：

```
var a = new String(1);
var b = new Boolean(true);
a = a + "";
b = b + "";
alert(a);                     // 返回字符串"1"
alert(b);                     // 返回字符串"true"
```

- 当对象与数值进行比较运算时，则会尝试把对象转换为数值，然后参与比较运算。如果不能够转换为有效数值，则执行字符串比较运算。例如：

```
var a = new String("true");   // 无法转换为数值
var b = new Boolean(true);    // 可以转换为数值 1
a = a > 0 ;
b = b > 0 ;
alert(a);                     // 返回 false，以字符串形式进行比较
```

```
alert(b);                    // 返回 true，以数值形式进行比较
```

- 当对象与字符串进行比较运算时，则直接转换为字符串，进行比较操作。

对于 Date 对象来说，加号运算符会先调用 toString()方法进行转换。因为当加号运算符作用于 Date 对象时，一般都是字符串连接操作。而当比较运算符作用于 Date 对象时，则会转换为数字以便比较时间的先后。