17.9.8 强制转换

JavaScript 支持使用下面方法强制类型转换。

- Boolean(value)：把参数值转换为 boolean 型。
- Number(value)：把参数值转换为 number 型。
- String(value)：把参数值转换为 string 型。

【示例】在下面代码中，分别调用上述三个函数，把参数值强制转换为新的类型值。

```
var a = String(true);          // 返回字符串"true"
var a = String(0);             // 返回字符串"0"
var b = Number("1");           // 返回数值 1
var c = Number(true);          // 返回数值 1
var d = Number("a");           // 返回 NaN
var e = Boolean(1);            // 返回 true
var f = Boolean("");           // 返回 false
```

【注意】

使用强制方式转换数据类型，有时候会产生异想不到的情况。例如，在上面示例中，true 被强制转换为数值 1，Number(false)会转换为 0，而使用 parseInt()方法转换时，它们都返回 NaN。

```
var a = parseInt(true);        // 返回 NaN
var b = parseInt(false);       // 返回 NaN
```

当要转换的值是至少有一个字符的字符串、非 0 数字或对象时，Boolean()函数将返回 true。如果该值是空字符串、数字 0、undefined 或 null，则它将返回 false。

Number()函数的强制类型转换与 parseInt()和 parseFloat()方法的处理方式相似，只是它转换的是整个值，而不是部分值。例如：

```
var a = Number("123abc");      // 返回 NaN
var b = parseInt("123abc");    // 返回数值 123
```

String()函数与 toString()方法功能基本相同。但是 String()函数能够把 null 或 undefined 值强制转换为字符串，而不引发错误。例如：

```
var a = String(null);          // 返回字符串"null"
var b = String(undefined);     // 返回字符串"undefined"
```

但是下面用法都将导致异常：

```
var a = null.toString();
var b = undefined.toString();
```

在 JavaScript 中，使用强制类型转换有时候会非常有用，但是应该确保转换值的正确。