基本包装类型

为了便于操作基本类型值，ECMAScript 还提供了 3 个特殊的引用类型：Boolean、Number 和 String。这些类型与本章介绍的其他引用类型相似，但同时也具有与各自的基本类型相应的特殊行为。实际上，每当读取一个基本类型值的时候，后台就会创建一个对应的基本包装类型的对象，从而让我们能够调用一些方法来操作这些数据。来看下面的例子。

(1) 创建 String 类型的一个实例；
(2) 在实例上调用指定的方法；
(3) 销毁这个实例。
可以将以上三个步骤想象成是执行了下列 ECMAScript 代码。

```
var s1 = new String("some text");
var s2 = s1.substring(2);
s1 = null;
```

引用类型与基本包装类型的主要区别就是对象的生存期。使用 new 操作符创建的引用类型的实例，在执行流离开当前作用域之前都一直保存在内存中。而自动创建的基本包装类型的对象，则只存在于一行代码的执行瞬间，然后立即被销毁。这意味着我们不能在运行时为基本类型值添加属性和方法。来看下面的例子：

```
var s1 = "some text";
s1.color = "red";
alert(s1.color);    //undefined
```

要注意的是，使用 new 调用基本包装类型的构造函数，与直接调用同名的转型函数是不一样的。例如：

```
var value = "25";
var number = Number(value);  //转型函数
alert(typeof number);        //"number"

var obj = new Number(value); //构造函数
alert(typeof obj);           //"object"
```