基本包装类型

> ! 本文主要介绍JavaScript中的基本包装类型以及使用注意点。

基本包装类型介绍

基本(简单)数据类型 字符串 + 数值 + null + undefined + 布尔值

为了便于操作基本类型，ECMAScript提供了三个特殊的引用类型： Boolean 、 Number 以及 String 类型，它们的结构与其他的引用类型(譬如 Array Object)类似。它们具备与各自的基本类型相应的特殊行为，每当我们读取一个基本类型的值的时候，后台就会创建一个对应的基本包装类型对象，从而让我们能够调用一些方法来操作这些数据。

```
var str = '测试字符串';
console.log(str.length);            //5
console.log(str.substring(2));      //字符串
```

在这里我们需要思考一个非常严肃而重要的问题，那就是属性和方法本是对象的特征，字符串如何能够拥有 length 属性以及其他类似 subString 等方法，内部怎么实现的?

HTML

我们知道基本类型值它们本身并不是对象，因此从逻辑上讨论他们不应该有属性和方法。
它们能够访问属性和方法的原理是在执行相关代码的时候，内部会执行下面的操作：

(1) 创建String类型的一个实例对象
(2) 在实例对象上面读取指定的属性（length），调用指定的方法（subString）
(3) 销毁该对象

基本包装类型的成员

```
/*01-创建Number类型的对象*/
var num = new Number(10);

/*02-创建String类型的对象*/
```

```
var str = new String('hello World');

/*03-创建Boolean类型的对象*/
var bool = new Boolean(true);
```

String	String 是与字符串(`string`)相对应的对象类型。
Number	Number 是与数字值(`number`)相对应的对象类型。
Boolean	Boolean 是与布尔值(`boolean`)相对应的对象类型。

```
//001 String
var str = '测试字符串';
console.log(str.length);              //5
console.log(str.substring(2));        //字符串

//002 Number
var num = new Number(10);
console.log(num);                     //Number {[[PrimitiveValue]]: 10}
console.log(typeof num);              //object
console.log(typeof 10);               //number

//003 Boolean
var bool = new Boolean(true);
console.log(bool);                    //Boolean {[[PrimitiveValue]]: true}
console.log(typeof bool);             //object
console.log(typeof true);             //boolean
```

基本包装类型的注意点

① 区分对象和基本值

对象 通过 new 调用构造函数(String、Boolean和Number)创建出来的是对象。

基本值 直接通过字面量方式赋值 或者 通过省略new关键字直接调用构造函数方式创建的基本值。

```
 var str1 = new String('hello');
 var str2 = 'hello';
 var str3 = String('hello');

/*说明：str1是对象，而str2和str3是字符串（基本数据类型值）*/
```

② 相等问题

```
var str1 = '这是一个字符串';                 //基本数据类型
var str2 = String('这是一个字符串');         //基本数据类型
console.log(str1 == str2);                //true 相等


var str3 = new String('这是一个字符串');     //引用类型-对象
console.log(str1 == str3);     //true     //值相等
console.log(str2 == str3);     //true     //值相等
```

```
console.log(str1 === str3);    //false  //值相等,但是引用不相等
console.log(str2 === str3);    //false  //值相等,但是引用不相等

/*判断下面的变量是否相等*/
var num1 = 10;                          //基本数据类型
var num2 = new Number(10);              //对象
console.log(num1 == num2);              //true
console.log(num1 ===  num2);            //false

var bool1 = true;
var bool2 = new Boolean(true);
console.log(bool1 == bool2);            //true
console.log(bool1 === bool2);           //false
```

基本类型值相等 => **值相等**

引用类型值相等 => **值相等且引用相等**

! 对象是引用类型，因此在判断相等的时候有诸多的注意点和容易出错的地方。