

第 8 章 对象和数组

学习要点:

- 1.Object 类型
- 2.Array 类型
- 3.对象中的方法

主讲教师: 李炎恢

合作网站: <http://www.ibeifeng.com>

讲师博客: <http://hi.baidu.com/李炎恢>

什么是对象, 其实就是一种类型, 即引用类型。而对象的值就是引用类型的实例。在 ECMAScript 中引用类型是一种数据结构, 用于将数据和功能组织在一起。它也常被称做为类, 但 ECMAScript 中却没有这种东西。虽然 ECMAScript 是一门面向对象的语言, 却不具备传统面向对象语言所支持的类和接口等基本结构。

一. Object 类型

到目前为止, 我们使用的引用类型最多的可能就是 Object 类型了。虽然 Object 的实例不具备多少功能, 但对于在应用程序中的存储和传输数据而言, 它确实是非常理想的选择。

创建 Object 类型有两种。一种是使用 new 运算符, 一种是字面量表示法。

1.使用 new 运算符创建 Object

```
var box = new Object();           //new 方式
box.name = '李炎恢';             //创建属性字段
box.age = 28;                    //创建属性字段
```

2.new 关键字可以省略

```
var box = Object();              //省略了 new 关键字
```

3.使用字面量方式创建 Object

```
var box = {                      //字面量方式
  name: '李炎恢',               //创建属性字段
  age: 28
};
```

4.属性字段也可以使用字符串星矢

```
var box = {
  'name': '李炎恢',             //也可以用字符串形式
  'age': 28
};
```

5.使用字面量及传统复制方式

```
var box = {};                   //字面量方式声明空的对象
```

```
box.name = '李炎恢';           //点符号给属性复制
box.age = 28;
```

6.两种属性输出方式

```
alert(box.age);               //点表示法输出
alert(box['age']);            //中括号表示法输出，注意引号
```

PS: 在使用字面量声明 Object 对象时，不会调用 Object()构造函数(Firefox 除外)。

7.给对象创建方法

```
var box = {
    run : function () {        //对象中的方法
        return '运行';
    }
}
alert(box.run());              //调用对象中的方法
```

8.使用 delete 删除对象属性

```
delete box.name;               //删除属性
```

在实际开发过程中，一般我们更加喜欢字面量的声明方式。因为它清晰，语法代码少，而且还给人一种封装的感觉。字面量也是向函数传递大量可选参数的首选方式。

```
function box(obj) {            //参数是一个对象
    if (obj.name !== undefined) alert(obj.name); //判断属性是否存在
    if (obj.age !== undefined) alert(obj.age);
}

box({                            //调用函数传递一个对象
    name : '李炎恢',
    age : 28
});
```

二. Array 类型

除了 Object 类型之外，Array 类型是 ECMAScript 最常用的类型。而且 ECMAScript 中的 Array 类型和其他语言中的数组有着很大的区别。虽然数组都是有序排列，但 ECMAScript 中的数组每个元素可以保存任何类型。ECMAScript 中数组的大小也是可以调整的。

创建 Array 类型有两种方式：第一种是 new 运算符，第二种是字面量。

1.使用 new 关键字创建数组

```
var box = new Array();          //创建了一个数组
var box = new Array(10);        //创建一个包含 10 个元素的数组
var box = new Array('李炎恢',28,'教师','盐城'); //创建一个数组并分配好了元素
```

2.以上三种方法，可以省略 new 关键字。

```
var box = Array();
```

//省略了 new 关键字

3 使用字面量方式创建数组

```
var box = [];
```

//创建一个空的数组

```
var box = ['李炎恢',28,'教师','盐城'];
```

//创建包含元素的数组

```
var box = [1,2,];
```

//禁止这么做，IE 会识别 3 个元素

```
var box = [,,,,];
```

//同样，IE 的会有识别问题

PS: 和 Object 一样，字面量的写法不会调用 Array()构造函数。(Firefox 除外)。

4.使用索引下标来读取数组的值

```
alert(box[2]);
```

//获取第三个元素

```
box[2] = '学生';
```

//修改第三个元素

```
box[4] = '计算机编程';
```

//增加第五个元素

5.使用 length 属性获取数组元素量

```
alert(box.length)
```

//获取元素个数

```
box.length = 10;
```

//强制元素个数

```
box[box.length] = 'JS 技术';
```

//通过 length 给数组增加一个元素

6.创建一个稍微复杂一点的数组

```
var box = [
```

```
{
```

//第一个元素是一个对象

```
    name : '李炎恢',
```

```
    age : 28,
```

```
    run : function () {
```

```
        return 'run 了';
```

```
    }
```

```
},
```

```
['马云','李彦宏',new Object()];
```

//第二个元素是数组

```
'江苏',
```

//第三个元素是字符串

```
25+25,
```

//第四个元素是数值

```
new Array(1,2,3)
```

//第五个元素是数组

```
];
```

```
alert(box);
```

PS: 数组最多可包含 4294967295 个元素，超出即会发生异常。

三. 对象中的方法

转换方法

对象或数组都具有 `toLocaleString()`、`toString()` 和 `valueOf()` 方法。其中 `toString()` 和 `valueOf()` 无论重写了谁，都会返回相同的值。数组会讲每个值进行字符串形式的拼接，以逗号隔开。

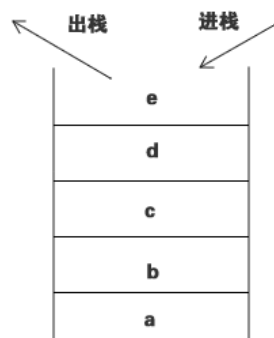
```
var box = ['李炎恢', 28, '计算机编程']; //字面量数组
alert(box); //隐式调用了 toString()
alert(box.toString()); //和 valueOf() 返回一致
alert(box.toLocaleString()); //返回值和上面两种一致
```

默认情况下，数组字符串都会以逗号隔开。如果使用 `join()` 方法，则可以使用不同的分隔符来构建这个字符串。

```
var box = ['李炎恢', 28, '计算机编程'];
alert(box.join("|")); //李炎恢|28|计算机编程
```

栈方法

ECMAScript 数组提供了一种让数组的行为类似于其他数据结构的方法。也就是说，可以让数组像栈一样，可以限制插入和删除项的数据结构。栈是一种数据结构(后进先出)，也就是说最新添加的元素最早被移除。而栈中元素的插入(或叫推入)和移除(或叫弹出)，只发生在一个位置——栈的顶部。ECMAScript 为数组专门提供了 `push()` 和 `pop()` 方法。

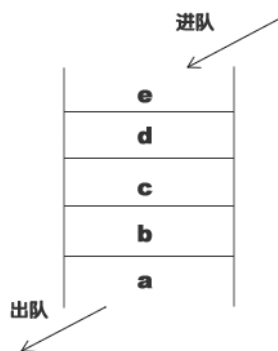


`push()` 方法可以接收任意数量的参数，把它们逐个添加到数组的末尾，并返回修改后数组的长度。而 `pop()` 方法则从数组末尾移除最后一个元素，减少数组的 `length` 值，然后返回移除的元素。

```
var box = ['李炎恢', 28, '计算机编程']; //字面量声明
alert(box.push('盐城')); //数组末尾添加一个元素，并且返回长度
alert(box); //查看数组
box.pop(); //移除数组末尾元素，并返回移除的元素
alert(box); //查看元素
```

队列方法

栈方法是后进先出，而队列方法就是先进先出。队列在数组的末端添加元素，从数组的前端移除元素。通过 `push()` 向数组末端添加一个元素，然后通过 `shift()` 方法从数组前端移除一个元素。



```
var box = ['李炎恢', 28, '计算机编程']; // 字面量声明
alert(box.push('盐城')); // 数组末尾添加一个元素，并且返回长度
alert(box); // 查看数组
alert(box.shift()); // 移除数组开头元素，并返回移除的元素
alert(box); // 查看数组
```

ECMAScript 还为数组提供了一个 `unshift()` 方法，它和 `shift()` 方法的功能完全相反。
`unshift()` 方法为数组的前端添加一个元素。

```
var box = ['李炎恢', 28, '计算机编程']; // 字面量声明
alert(box.unshift('盐城', '江苏')); // 数组开头添加两个元素
alert(box); // 查看数组
alert(box.pop()); // 移除数组末尾元素，并返回移除的元素
alert(box); // 查看数组
```

PS: IE 浏览器对 `unshift()` 方法总是返回 `undefined` 而不是数组的新长度。

重排序方法

数组中已经存在两个可以直接用来排序的方法：`reverse()` 和 `sort()`。

`reverse()` 逆向排序

```
var box = [1,2,3,4,5]; // 数组
alert(box.reverse()); // 逆向排序方法，返回排序后的数组
alert(box); // 源数组也被逆向排序了，说明是引用
```

`sort()` 从小到大排序

```
var box = [4,1,7,3,9,2]; // 数组
alert(box.sort()); // 从小到大排序，返回排序后的数组
alert(box); // 源数组也被从小到大排序了
```

`sort` 方法的默认排序在数字排序上有些问题，因为数字排序和数字字符串排序的算法是一样的。我们必须修改这一特征，修改的方式，就是给 `sort(参数)` 方法传递一个函数参数。这点可以参考手册说明。

```
function compare(value1, value2) { // 数字排序的函数参数
    if (value1 < value2) { // 小于，返回负数
        return -1;
    } else if (value1 > value2) { // 大于，返回正数
```

```
        return 1;
    } else {
        return 0;
    }
    var box = [0,1,5,10,15];
    alert(box.sort(compare));
```

//其他, 返回 0

//验证数字字符串, 和数字的区别

//传参

PS: 如果要反向操作, 即从大到小排序, 正负颠倒即可。当然, 如果要逆序用 `reverse()` 更加方便。

操作方法

ECMAScript 为操作已经包含在数组中的元素提供了很多方法。`concat()`方法可以基于当前数组创建一个新数组。`slice()`方法可以基于当前数组获取指定区域元素并创建一个新数组。`splice()`主要用途是向数组的中部插入元素。

```
var box = ['李炎恢', 28, '盐城'];
var box2 = box.concat('计算机编程');
alert(box2);
alert(box);
```

//当前数组

//创建新数组, 并添加新元素

//输出新数组

//当前数组没有任何变化

```
var box = ['李炎恢', 28, '盐城'];
var box2 = box.slice(1,3);
alert(box2);
alert(box);
```

//当前数组

//box.slice(1,3), 2-4 之间的元素

//28, 盐城

//当前数组

`splice` 中的删除功能:

```
var box = ['李炎恢', 28, '盐城'];
var box2 = box.splice(0,2);
alert(box2);
alert(box);
```

//当前数组

//截取前两个元素

//返回截取的元素

//当前数组被截取的元素被删除

`splice` 中的插入功能:

```
var box = ['李炎恢', 28, '盐城'];
var box2 = box.splice(1,0,'计算机编程','江苏');
alert(box2);
alert(box);
```

//当前数组

//没有截取, 但插入了两条

//在第 2 个位置插入两条

//输出

`splice` 中的替换功能:

```
var box = ['李炎恢', 28, '盐城'];
var box2 = box.splice(1,1,100);
alert(box2);
alert(box);
```

//当前数组

//截取了第 2 条, 替换成 100

//输出截取的 28

//输出数组

感谢收看本次教程！

本课程是由北风网(ibeifeng.com)

瓢城 **Web** 俱乐部(yc60.com)联合提供：

本次主讲老师：李炎恢

我的博客：hi.baidu.com/李炎恢/

我的邮件：yc60.com@gmail.com