js函数中有三个在特定场合很好用的函数:reduce(),map(),filter()。而数组经常用到的方法有push、join、indexOf、slice等等，但是有一个经常被我们忽略的方法：reduce，这个方法简直强大的不要不要的。下面通过这篇文章来一起学习学习吧。

## reduce简介

我们先来看看这个方法的官方概述：reduce() 方法接收一个函数作为累加器（accumulator），数组中的每个值（从左到右）开始缩减，最终为一个值。

你一定也和我一样看的有点迷糊，其实reduce接收的就是一个回调函数，去调用数组里的每一项，直到数组结束。

**我们来举个例子大家就很明白了。**

假设我有一串数组，数组里放的全是数字，我要算出这些数字的总和是多少。正常情况下我们会循环，然后一个个加，有了reduce就不用那么麻烦了，只用一行代码。

|  |  |
| --- | --- |
| 1 | var total = [0,1,2,3,4].reduce((a, b)=>a + b); //10 |

**这个方法是如何工作的呢？**

reduce接受一个函数，函数有四个参数，分别是：

     1、上一次的值;

     2、当前值;

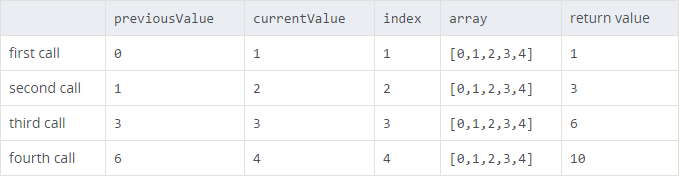
     3、当前值的索引;

     4、数组;

我们还是以上面那个数组为例，把这几个参数打印出来看看:

|  |  |
| --- | --- |
| 1  2  3 | [0, 1, 2, 3, 4].reduce(function(previousValue, currentValue, index, array){   return previousValue + currentValue;  }); |

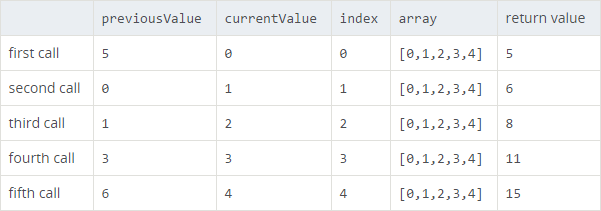
得到的结果是：

[](http://files.jb51.net/file_images/article/201612/201612294807546.png?201611294816)

分析一下这个结果，这个回调函数共调用了4次，因为第一次没有previousValue，所以直接从数组的第二项开始，一只调用到数组结束。

reduce还有第二个参数，我们可以把这个参数作为第一次调用callback时的第一个参数，上面这个例子因为没有第二个参数，所以直接从数组的第二项开始，如果我们给了第二个参数为5，那么结果就是这样的：

|  |  |
| --- | --- |
| 1  2  3 | [0, 1, 2, 3, 4].reduce(function(previousValue, currentValue, index, array){   return previousValue + currentValue;  },5); |



第一次调用的previousValue的值就用传入的第二个参数代替，函数被调用了5次，也就是数组的length。

## reduce作用：

reduce可以帮助我们轻松的完成很多事，除了累加，还有扁平化一个二维数组：

|  |  |
| --- | --- |
| 1  2  3  4 | var flattened = [[0, 1], [2, 3], [4, 5]].reduce(function(a, b) {   return a.concat(b);  }, []);  // flattened == [0, 1, 2, 3, 4, 5] |

--------------------------------------------------------------------------------------------------------------------------------------------------------------

# Object.assign()解释整理

<http://blog.csdn.net/wang252949/article/details/79106160>

--------------------------------------------------------------------------------------------------------------------------------------------------------------

**Object.keys()** 方法会返回一个由一个给定**对象**的自身可枚举属性组成的**数组**，数组中属性名的排列顺序和使用 [for...in](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Statements/for...in) 循环遍历该对象时返回的顺序一致 （两者的主要区别是 一个 for-in 循环还会枚举其原型链上的属性）。

1. Object.keys() 和 for...in 输出顺序一样
2. 输出顺序规则为：
3. 按（integer indices）升序输出
4. 按添加顺序输出字符串key
5. 按添加顺序输出Symbol key （es6 中新加的）

integer indices ： 是指将key转化为53bit的无符号整型，然后再转会字符串 后依旧相等的 key。如 '10', '2' 为integer indices。 '01', '12a' 不是 integer indices。