## 1：ES5简介

ECMAScript 5.1 (或仅 ES5) 是ECMAScript(基于JavaScript的规范)标准的修正。 与HTML5规范进程本质类似，ES5通过对现有JavaScript方法添加语句和原生ECMAScript对象做合并实现标准化。

## 2：严格模式简介

除了正常运行模式，ECMAscript 5添加了第二种运行模式："严格模式"（strict mode）。顾名思义，这种模式使得Javascript在更严格的条件下运行。

设立"严格模式"的目的，主要有以下几个：

1：消除Javascript语法的一些不合理、不严谨之处，减少一些怪异行为。

2：消除代码运行的一些不安全之处，保证代码运行的安全:。

3：提高编译器效率，增加运行速度。

4：为未来新版本的Javascript做好铺垫。

"严格模式"体现了Javascript更合理、更安全、更严谨的发展方向，包括IE 10在内的主流浏览器，都已经支持它，许多大项目已经开始全面拥抱它。

## 3：进入严格模式

进入“严格模式”的标志，书写这行语句："use strict";

老版本的浏览器会把它当作一行普通字符串，加以忽略。

## 4：如何调用严格模式

"严格模式"有两种调用方法，适用于不同的场合。

**针对整个脚本文件**：将"use strict"放在脚本文件的第一行，则整个脚本都将以"严格模式"运行。如果这行语句不在第一行，则无效，整个脚本以"正常模式"运行。如果不同模式的代码文件合并成一个文件，这一点需要特别注意。

**针对单个函数**：将"use strict"放在函数的第一行，则整个函数以"严格模式"运行。

**脚本文件的变通写法：**因为第一种调用方法不利于文件合并，所以更好的做法是，借用第二种方法，将整个脚本文件放在一个立即执行的匿名函数之中。

## 5：严格模式改变语法和行为

严格模式对Javascript的语法和行为，都做了一些改变。

**1：全局变量显式声明。**

在正常模式中，如果一个变量没有声明就赋值，默认是全局变量。严格模式禁止这种用法，全局变量必须显式声明。因此，严格模式下，变量都必须先用var命令声明，然后再使用。

**2：函数形参不允许重复。**

**3：禁止使用with语句。**

**4：禁止函数内部this关键字指向全局对象，this指向undefined。**

**5：函数必须声明在顶层。**

将来Javascript的新版本会引入“块级作用域"。为了与新版本接轨，严格模式只允许在全局作用域或函数作用域的顶层声明函数。也就是说，不允许在非函数的代码块内声明函数。

// if(1){function a(){}} else {function a() {}}

## 6：ES5新增函数方法

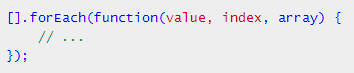
注：新增方法IE6/7/8均不支持。

**1：bind方法**：改变了函数内部上下文的this。返回值是函数。在调用返回函数时，传入的实参是在bind中传入参数的基础上往后排的。



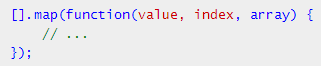
## 7：ES5新增数组方法

**1：forEach方法：**遍历数组。第二个参数为可选的上下文参数（改变this指向）（返回原数组）

forEach方法中的function回调支持3个参数，第1个是遍历的数组内容；第2个是对应的数组索引，第3个是数组本身。forEach不会遍历空位置的内容。如：[1, , 3]，只会遍历1和3。

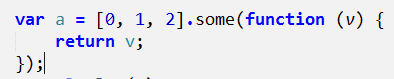
**2：map方法**：将数组映射成新数组。回调函数需要有返回值。

**3：filter方法：**指数组过滤后，返回过滤后的新数组。 filter的callback函数需要返回布尔值true或false。

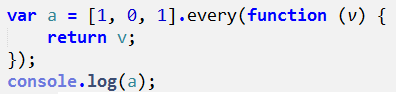


**4：some方法：**指是否“某些项”合乎条件。返回布尔值。只要数组中有一项符合条件，则返回true。

**5：every方法**：所有项都必须合乎条件。返回布尔值。

语法：array.every(callback, [ thisobject]); 必须每一项都要符合条件，否则返回false。



**6：indexOf方法：**返回整数索引值，如果没有匹配（严格匹配），返回-1。 fromIndex可选，表示从这个位置开始搜索，若缺省或格式不合要求，使用默认值0。



**7：lastIndexOf方法**：从数组的末尾开始查找，而不是从开头。还有一个不同就是fromIndex的默认值是array.length - 1而不是0。

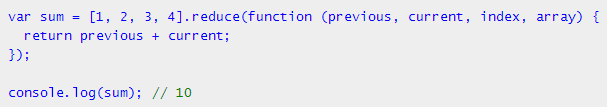


**8：reduce方法：归并。递归减少。**

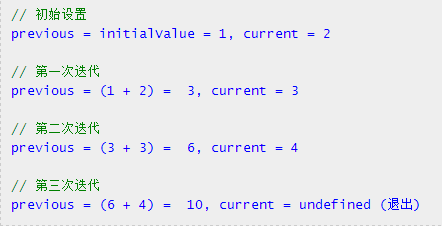
callback函数接受4个参数：之前值、当前值、索引值以及数组本身。initialValue参数可选，表示初始值。若指定，则当作最初使用的previous值；如果缺省，则使用数组的第一个元素作为previous初始值，同时current往后排一位，相比有initialValue值少一次迭代。



案例：



解析：



说明：

1：因为initialValue不存在，因此一开始的previous值等于数组的第一个元素。

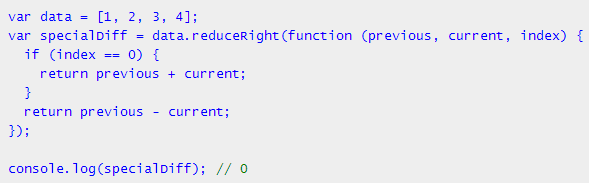
2：从而current值在第一次调用的时候就是2。

3：最后两个参数为索引值index以及数组本身array。

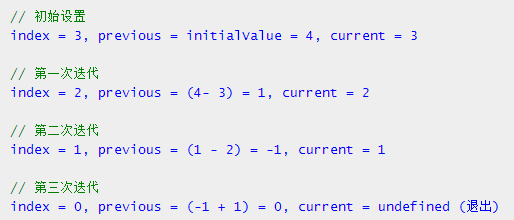
**9：reduceRight方法**：和reduce用法类似，实现上差异在于reduceRight是从数组的末尾开始实现。



案例：



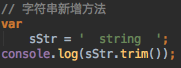
说明：



**10：Array.isArray()**：判断一个变量是否为数组。

## 8：ES5新增字符串方法

**1：trim方法：**去除字符串首尾的空白字符。



## 应用：

1：预习下午的字符串。

2：背诵记忆ES5。

## 1：字符串

JS中的任何数据类型都可以当作对象来看。所以string既是基本数据类型，又是对象。

## 2：声明字符串

var sStr = ‘字符串’;

var oStr = new String(‘字符串’);

## 3：字符串属性

1.length

计算字符串的长度(不区分中英文)。

2.constructor

对象的构造函数。

## 4：字符串方法

1.myStr.charAt(num)

返回在指定位置的字符。

2.myStr.charCodeAt(num)

返回指定位置的字符的Unicode（是字符编码的一种模式）编码。

3.String.fromCharCode(num1,num2,num3,…)

String的意思就是不能用自己定义的字符串名字来调用，例如定义一个变量字符串 var str="";只能用String来定义调用。接受一个或多个指定的Unicode值，然后返回一个或多个字符串。(把unicode编码转换为字符串)。

4.myStr.indexOf()

返回某个指定的字符串，在字符串中首次出现的位置。如果要检索的字符串值没有出现，则该方法返回 -1。

5.myStr.lastIndexOf()

返回一个指定的字符串值最后出现的位置，如果要检索的字符串值没有出现，则该方法返回 -1。

6.myStr.match()

在字符串中检索指定的值，返回的值是数组。如果匹配不到返回null。配合正则来用。

7.myStr.search()

返回出现的位置，查找不到返回-1。配合正则来用。

8.myStr.replace(“需替换的字符串”,“替换后的字符串”)

将字符串中的一些字符替换为另外一些字符。配合正则使用。

9.myStr.slice(start,end)

从指定的开始位置，到结束位置(不包括结束位置)的所有字符串。如果不指定结束位置，则从指定的开始位置，取到结尾。注意的是，myStr.slice() 与 myArr.slice() 相似。

10.myStr.substring(start,end)

从指定的开始位置，到结束位置(不包括)的所有字符串。如果不指定结束位置，则从指定的开始位置，取到结尾。

11.substr(start,length)

从指定的位置开始取指定长度的字符串。如果没有指定长度，从指定开始的位置取到结尾。 ECMAscript 没有对该方法进行标准化，**因此反对使用它**。

如果substr的start指定为负数，则该参数声明从字符串的尾部开始算起的位置。也就是说，-1 指字符串中最后一个字符，-2 指倒数第二个字符，以此类推。

—————————— slice(start,end) vs substring(start,end) ————————————

slice参数可以是负数，如果是负数，从-1开始指的是字符串结尾。

substring参数是负数的时候，会自动转换为0。

——————————————————————————————————————————————

12.split("分割位置",[指定的长度])

将一个字符串分割成数组。

13.toLowerCase()

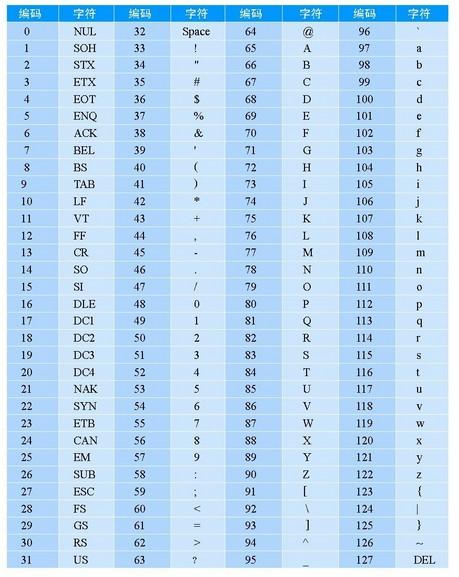
用于把字符串转换为小写。

14.toUpperCase()

将字符串转换为大写。

## 5：ASCII码和字符集

ASCII：American Standard Code for Information Interchange，美国信息交换标准代码。



**Unicode编码：**

Unicode（统一码、万国码、单一码）是计算机科学领域里的一项业界标准,包括字符集、编码方案等。Unicode 是为了解决传统的字符编码方案的局限而产生的，它为每种语言中的每个字符设定了统一并且唯一的二进制编码，以满足跨语言、跨平台进行文本转换、处理的要求。 Unicode目前普遍采用的是UCS-2,它用两个字节来编码一个字符。

如汉字"经"的编码是0x7ECF,注意字符码一般用十六进制来 表示，为了与十进制区分，十六进制以0x开头，0x7ECF转换成十进制 就是32463,UCS-2用两个字节来编码字符，两个字节就是16位二进制， 2的16次方等于65536,所以UCS-2最多能编码65536个字符。

**GBK编码：**

GBK全称《汉字内码扩展规范》（GBK即“国标”、“扩展”汉语拼音的第一个字母，英文名称：Chinese Internal Code Specification）。GBK 向下与GB2312编码兼容，向上支持 ISO 10646.1国际标准，是前者向后者过渡过程中的一个承上启下的产物。

**UTF-8编码：**

UTF-8（8-bit Unicode Transformation Format）是一种针对Unicode的可变长度字符编码，又称万国码。 UTF-8用1到4个字节编码UNICODE字符。用在网页上可以同一页面显示中文简体繁体及其它语言（如英文，日文，韩文）。

## 应用：

1．敏感词过滤。

2．密码格式要求。

3．留言过滤。



## 综合应用：

1：统计字符串中每个字符的个数。

2：统计每个字符出现的次数，结果显示 a 2、b 1、c 2、d1，去掉重复的字符，使结果显示 abcdfgj。