logo.png

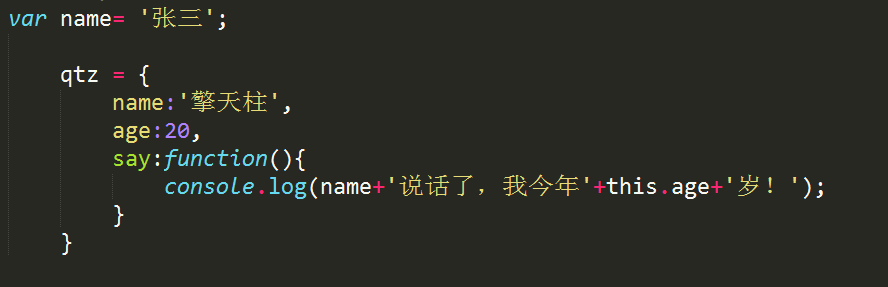
javascript基础五

H5 web前端

[www.bufanui.com](http://www.bufanui.com)

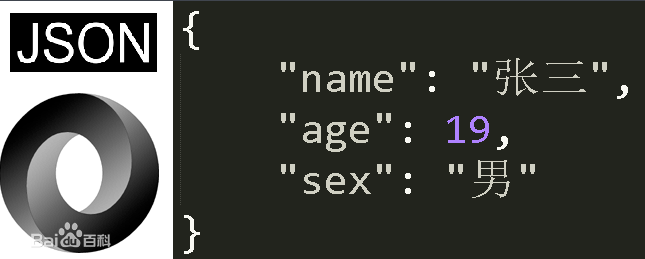
# 对象和Json

## 对象的表示方法



## 1.2 JSON

什么是json？ 键值对



[JSON](http://baike.baidu.com/item/JSON)([JavaScript](http://baike.baidu.com/item/JavaScript) Object Notation, JS 对象标记) 是一种轻量级的数据交换格式。它基于 [ECMAScript](http://baike.baidu.com/item/ECMAScript" \t "_blank) (w3c制定的js规范)的一个子集，采用完全独立于编程语言的文本格式来存储和表示数据。简洁和清晰的层次结构使得 JSON 成为理想的数据交换语言。 易于人阅读和编写，同时也易于机器解析和生成，并有效地提升网络传输效率。

之前使用的是xml传递数据，现在流行使用json



{

"q":"chengxuyuan",

"p":false,

"bs":"",

"csor":"0",

"status":0,"

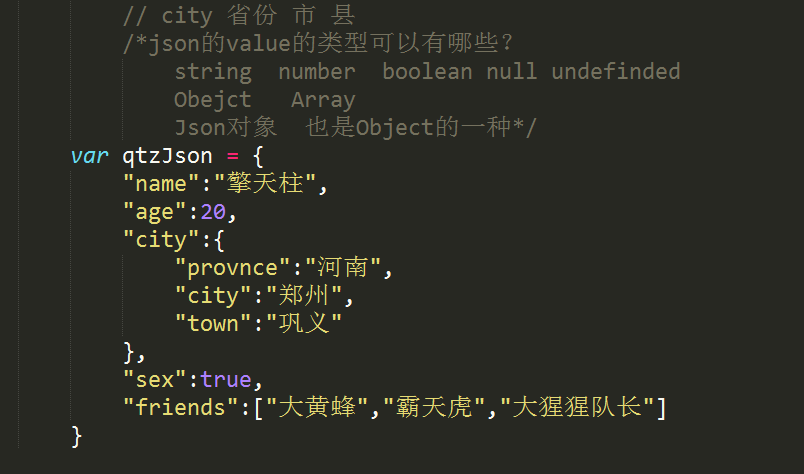
…}

拿到百度的数据之后 通过js进行解析 把解析出来的内容设置到页面中

## 1.3 Json语法规则

1.3.1 在 JS 语言中，一切都是对象。因此，任何支持的类型都可以通过 JSON 来表示，例如字符串、数字、对象、数组等。但是对象和数组是比较特殊且常用的两种类型：

* 对象表示为键值对
* 数据由逗号分隔
* 花括号保存对象
* 方括号保存数组



### 1.3.2 JSON 键/值对

键/值对组合中的键名写在前面并用双引号 "" 包裹，使用冒号 : 分隔

{"firstName": "John"}

## 1.4 JSON 与 JS 对象

## 1.4.1 json与js对象的关系

JSON 是 JS 对象的字符串表示法，它使用文本表示一个 JS 对象的信息，本质是一个字符串。

比如：

对象 var user = {name:’zhangsan’,age:20}

json var user = ‘{“name”:”zhangsan”,”age”:20}’

## 1.4.2 json和js对象的相互转化

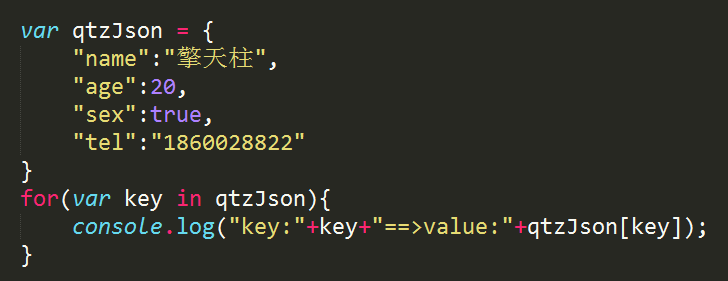
要实现从对象转换为 JSON 字符串，使用 **JSON.stringify()** 方法：

var json = JSON.stringify({name:’zhangsan’,age:20});

要实现从 JSON 转换为对象，使用 **JSON.parse()** 方法：

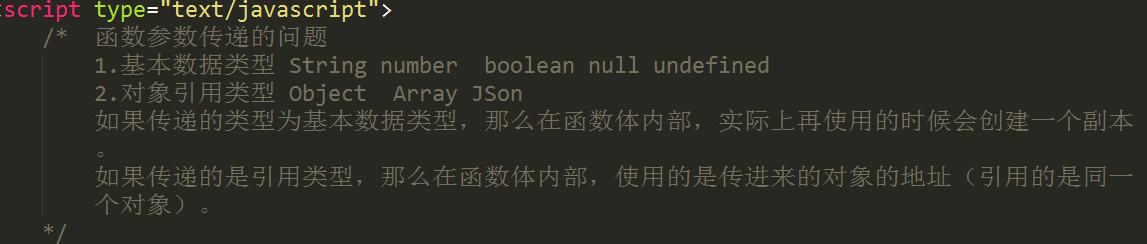
var user = JSON.parse(‘{“name”:”zhangsan”,”age”:20}’);

## 1.5 json的遍历



## 参数和传值问题

1. 简单类型数据做参数，函数内部对参数的修改不应影响外部变量
2. 复杂类型数据做参数，函数内部对参数的修改会应影响外部变量

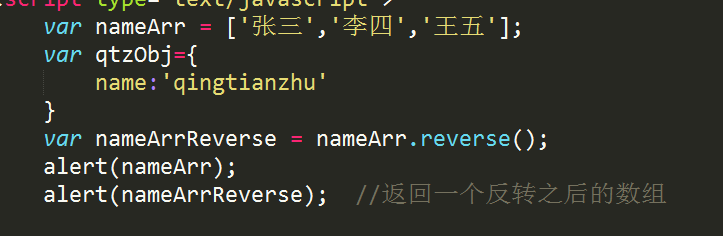


# 二、数组高级API

## Array的内置方法

**toString() //把数组转换成字符串，每一项用,分割**

**reverse() //翻转数组**

****

**sort(); //给数组排序，返回排序后的数组。如何排序看参数。**

无参：按照数组元素的首字符对应的Unicode编码值从小到大排列数组元素。

带参：必须为函数（回调函数--callback）。函数中带有两个参数，代表数组中的 前后元素。如果计算后（a-b），返回值为负数，a排b前面。等于0不动。 返回值为正数，a排b后面。



**indexOf()、lastIndexOf() //如果没找到返回-1**

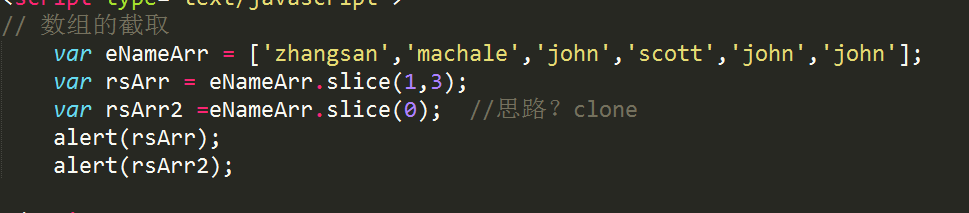
作业：var eNameArr = ['zhangsan','machale','john','scott','john','john'];

写个方法 要求 接受数组和字符串 取出该数组中所有该字符串的下标

### 了解方法

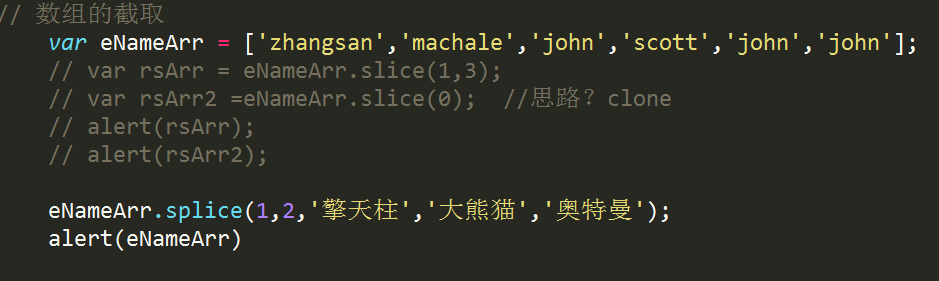
**slice() //从当前数组中截取一个新的数组，不影响原来的数组，参数start从0开始,end从1开始**

返回一个新的数组，包含从 start 到 end （不包括该元素）的 arrayObject 中的元素。



**splice()//删除或替换当前数组的某些项目，参数start,deleteCount,options(要替换的项目)**

arrayObject.splice(index,howmany,item1,.....,itemX)



### 清空数组

array.length = 0; //length属性可以赋值，其它语言中length是只读

array = []; //推荐

### 练习

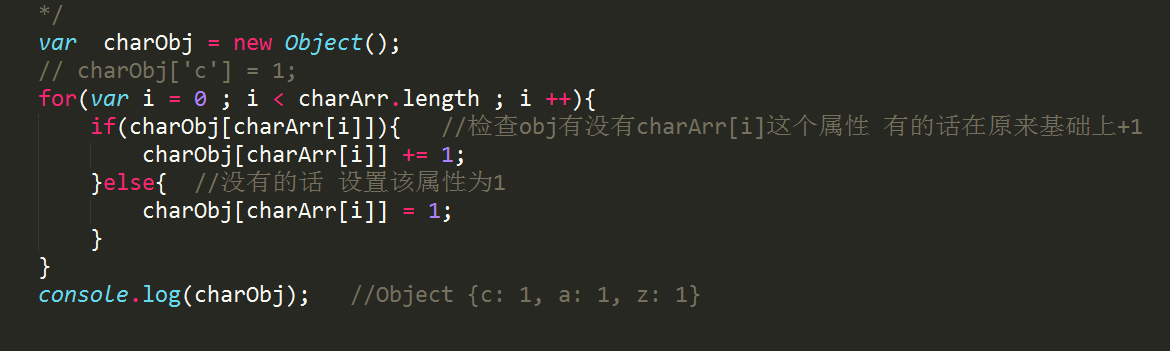
1.将一个字符串数组输出为|分割的形式，比如“刘备|张飞|关羽”。使用两种方式实现

2.将一个字符串数组的元素的顺序进行反转。["a","b","c","d"] ["d","c","b","a"]。使用两种种方式实现。

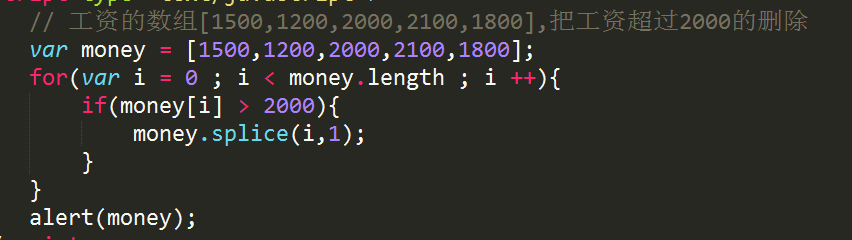
3.找到数组中每个字母出现的次数["c","a","z","a","a"] js没有map

# 如何把一个字符串的大小写取反（大写变小写小写变大写），例如 ’AbC' 变成 'aBc' 。

# 请把俩个数组 [A1, A2, B1, B2, C1, C2, D1, D2] 和 [A, B, C, D]，合并为 [A1, A2, A, B1, B2, B, C1, C2, C, D1, D2, D]。



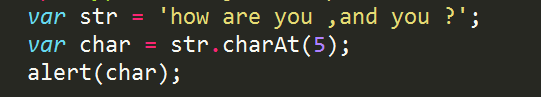
4.工资的数组[1500,1200,2000,2100,1800],把工资超过2000的删除



# 三、字符串对象的常用方法

## 3.1 常用方法

* charAt获取相应位置字符



* charCodeAt

charCodeAt() 方法可返回指定位置的字符的 Unicode 编码。这个返回值是 0 - 65535 之间的整数。

Unicode（[统一码](http://baike.baidu.com/view/3688170.htm)、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](http://baike.baidu.com/view/18536.htm" \t "_blank)编码，以满足跨语言、跨平台进行文本转换、处理的要求。1990年开始研发，1994年正式公布。

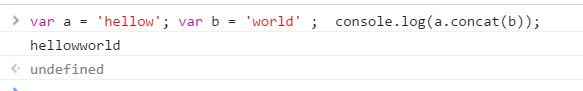
Unicode是国际组织制定的可以容纳世界上所有文字和符号的字符编码方案。目前的Unicode字符分为17组编排，0x0000 至 0x10FFFF，每组称为平面（Plane），而每平面拥有65536个码位，共1114112个。然而目前只用了少数平面。[UTF-8](http://baike.baidu.com/view/25412.htm" \t "_blank)、[UTF-16](http://baike.baidu.com/view/497266.htm" \t "_blank)、[UTF-32](http://baike.baidu.com/view/1453325.htm" \t "_blank)都是将数字转换到程序数据的编码方案。



* indexOf

返回字符在字符串中的位置

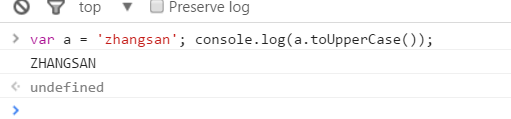
* lastindexOf
* concat() 连接字符串



* slice() 方法可提取字符串的某个部分，并以新的字符串返回被提取的部分。
* substr() 用的非常多

substr(起始位置,[取的个数])

* toUpperCase()



* toLowerCase()

## 3.2保留小数位数

Math.fixed();

附件：

**三种常见字符编码简介：ASCII、Unicode和UTF-8**

**什么是字符编码？**

      计算机只能处理数字，如果要处理文本，就必须先把文本转换为数字才能处理。最早的计算机在设计时采用8个比特（bit）（01010101）作为一个字节（byte），所以，一个字节能表示的最大的整数就是255（二进制11111111=十进制255），如果要表示更大的整数，就必须用更多的字节。比如两个字节可以表示的最大整数是65535，4个字节可以表示的最大整数是4294967295。

**ASCII编码：**

由于计算机是美国人发明的，因此，最早只有127个字符被编码到计算机里，也就是大小写英文字母、数字和一些符号，这个编码表被称为ASCII编码，比如大写字母 A 的编码是65，小写字母 z 的编码是122。

后来欧洲需要编码ascii不够 出现 iso-8859-1,到中国gbk，gb2312，big5

　　但是要处理中文显然一个字节是不够的，至少需要两个字节，而且还不能和ASCII编码冲突，所以，中国制定了GB2312编码，用来把中文编进去。

　　全世界有上百种语言，日本把日文编到Shift\_JIS里，韩国把韩文编到Euc-kr里，各国有各国的标准，就会不可避免地出现冲突，结果就是，在多语言混合的文本中，显示出来会有乱码。

**Unicode编码：**

　　因此，Unicode应运而生。Unicode把所有语言都统一到一套编码里，这样就不会再有乱码问题了。Unicode标准也在不断发展，但最常用的是用两个字节表示一个字符（如果要用到非常偏僻的字符，就需要4个字节）。现代操作系统和大多数编程语言都直接支持Unicode。

**ASCII编码和Unicode编码的区别：**

　　1）ASCII编码是1个字节，而Unicode编码通常是2个字节，举例如下。

　　字母 A 用ASCII编码是十进制的65，二进制的01000001；小写a是97

　　字符 0 用ASCII编码是十进制的48，二进制的00110000，注意字符 '0' 和整数 0 是不同的；

　　汉字 中 已经超出了ASCII编码的范围，用Unicode编码是十进制的20013，二进制的01001110 00101101。

　　如果把ASCII编码的 A 用Unicode编码，只需要在前面补0就可以，因此， A 的Unicode编码是00000000 01000001。

**UTF-8编码：**

　　新问题的出现：如果统一成Unicode编码，乱码问题从此消失了。但是，如果你写的文本基本上全部是英文的话，用Unicode编码比ASCII编码需要多一倍的存储空间，在存储和传输上就十分不划算。

　　因此，又出现了把Unicode编码转化为“可变长编码”的UTF-8编码。UTF-8编码把一个Unicode字符根据不同的数字大小编码成1-6个字节，常用的英文字母被编码成1个字节，汉字通常是3个字节，只有很生僻的字符才会被编码成4-6个字节。如果你要传输的文本包含大量英文字符，用UTF-8编码就能节省空间：

|  |  |  |  |
| --- | --- | --- | --- |
| **字符** | **ASCII** | **Unicode** | **UTF-8** |
| A | 01000001 | 00000000 01000001 | 01000001 |
| 中 | - | 01001110 00101101 | 11100100 10111000 10101101 |

　　从上面的表格可以发现UTF-8编码一个额外的好处，就是ASCII编码实际上可以被看成是UTF-8编码的一部分，所以，大量只支持ASCII编码的历史遗留软件可以在UTF-8编码下继续工作。