**数组(Array)**

一组数据就是数组，js的数组可以是任意数据类型，任意长度

——数组也是一个对象

——它和我们普通对象功能类似，也是用来存储一些值的

——不同的是普通对象是使用字符串作为属性名的.

而数组时使用数字来作为索引操作元素

——索引，是指从0开始的整数就是索引

——数组的存储性能比普通对象好，在开发中我们经常使用数组来存储一些数据

注意：不能用

var arr = [1,2,3];

var newarr = arr;这种方式是来复制一个一模一样的数组

因为变量中存放的是数组对象的一个地址 这种做法是将arr中存放的地址复制到newarr中

**数组操作**

**创建数组对象**

**1.使用构造函数创建一个Array类的对象arr**

var arr = new Array(10,20,30);

使用构造函数创建数组时，也可以同时添加元素，将要添加的元素作为构造函数的参数传递

——注意：如果只传递一个数字则表示数组的长度

元素之间使用,隔开

console.log(typeof arr);

使用typeof检查一个数组时会返回object

**2.使用字面量创建数组**

语法：[ ]

var arr = [ ];

使用字面量创建数组时，可以在创建时就指定数组中的元素

var arr = [1,2,3.4,5,10];

//创建一个数组数组中只有一个元素

arr = [10];

//创建一个长度为10的数组

arr2 = new Array ( 10);

**3.var arr = Array(100,200);**

**向数组中添加元素**

语法：

数组[索引]=值;

**读取数组中的元素**

语法：

数组[索引]=值;

如果读取不存在的索引,它不会报错而是返回undefined

**获取数组的长度**

可以使用length属性来获取数组的长度(元素的个数)

语法：

数组.length

对于连续的数组,使用length可以获取到数组的长度（元素的个数)

对于非连续的数组,使用length会获取到数组的最大的索引+1

尽量不要创建非连续的数组

**修改数组的长度**

数组的长度可以动态被修改

如果修改的length大于原长度，则多出部分会空着

如果修改的length小于原长度，则多出的元素会被删除

**length使用小技巧**

向数组的最后一个位置添加元素

语法：数组[数组,length]=值;

**数组的数据类型**

数组中的元素可以是任意的数据类型

arr = [ "hello",1,true, null, undefined];

也可以是对象

var obj = {name : "孙悟空"};

arr[arr.length] = obj;

arr = [{name: "孙悟空"},{name: "沙和尚"},{name: "猪八戒"}];

也可以是一个函数

arr =[function( ){alert(1)} , function( ){alert(2)}];

console.log( arr);

调用：arr[0]();

数组中也可以放数组，如下这种数组称为二维数组

arr = [[1,2,3],[3,4,5],[5,6,7]];

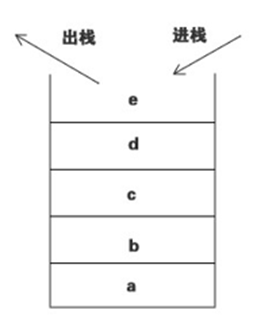
console.log(arr[1]);//输出1,2,3

**数组的方法**

**栈方法：后进先出 LIFO （Last In First Out）**

push+pop

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。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)，也就是说最后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()**

——该方法可以向数组的末尾逐个添加任意数量的元素，并返回修改后数组的新长度

——可以将要添加的元素作为方法的参数传递，

这样这些元素将会自动添加到数组的末尾

——该方法会将数组新的长度作为返回值返回

**使用：**

var result = arr.push("唐僧","蜘蛛精","白骨精","玉兔精");

**pop()**

——该方法可以删除数组的最后一个元素,并将被删除的元素作为返回值返回

**使用：**

result = arr.pop();

**队列方法：先进先出 FIFO （First In First Out）**

队列在数组的末端添加元素，从数组的前端移除元素

push+shift

**push():**

——和栈方法中的push是同一个

**unshift()**

——向数组开头添加一个或多个元素，并返回新的数组长度

——向前边插入元素以后，其他的元素索引会依次调整

**使用：**

arr.unshift("牛魔王","二郎神");

**shift()**

——可以删除数组的第一个元素，并将被删除的元素作为返回值返回

**使用：**

arr.shift();

**排序方法**

**reverse() :**

逆向排序

该方法用来反转数组〔前边的去后边，后边的去前边)

该方法会直接修改原数组

**语法：**

var arr = [1,2,3,4,5];

console.log(arr.reverse());

**sort() :**

用来对数组进行排序，也会直接修改原数组

默认会按照ASCII码进行升序排序

**语法:**

arr.sort();

即使是纯数字的数组，也会按照ASCII码进行升序排序，从第一位开始比较大小。

可以自己指定排序的规则：

给 sort()中添加一个回调函数，来指定排序规则。

回调函数需要定义两个形参，浏览器分别使用数组中的元素作为实参去调用回调函数。

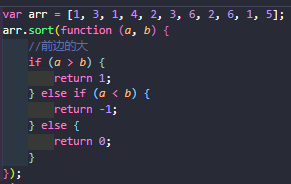
使用哪个元素调用不确定，只能确定实参a一定在实参b前面

浏览器会根据回调函数的返回值来决定元素的顺序,

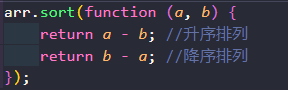
如果返回一个大于0的值，则元素会交换位置

如果返回一个小于0的值，则元素位置不变

如果返回一个0，则认为两个数相等，也不交换位置



简写：



**数组的方法（ES3）：**

**concat() : 1.数组的合并 2.数组的拷贝**

——可以追加数据, 创建一个新数组, 不改变原数组

——可以连接两个或多个数组,并将新的数组返回

**语法：**

//将arr数组追加两个元素得到arr2数组

var arr2 = arr.concat(‘美队’, ‘雷神’);

//将arr和arr2连接起来得到新数组用result存放

var result =arr.concat(arr2);

**slice() :**

将原数组中的指定区域数据提取出来，不改变原数组，理解为**复制**

该方法不会改变元素数组，而是将截取到的元素封装到一个新数组中返回

**语法：**

**参数:**

1.截取开始的位置的索引，包括开始索引

2.截取结束的位置的索引，不包括结束索引

——第二个参数可以省略不写，不写则表示截取第一个参数往后所有的元素

——如果传递一个负值，则从后往前计算，-1表示截取到倒数第一个(不包括)

**eg.**

var arr2 = arr.slice(1, 3);

**splice() :**

截取原数组中指定的数据, 会改变原数组

**删除：**

使用splice()会影响到原数组，会将指定元素从原数组中删除

并将被删除的元素作为返回值返回

**参数：**

第一个，表示开始位置的索引，包括开始的索引

第二个，表示删除的数量

**语法：**

var arr = [2, 3, 4, '绿箭侠', '黑寡妇'];

var arr2 = arr.splice(0, 3);

**插入：**

**参数：**

第一个，表示开始位置的索引

第二个，为0

第三个，要插入的新的元素，这些元素将会自动插入到开始位置索引前边

**语法：**

var arr = [2, 3, 4, '绿箭侠', '黑寡妇'];

arr.splice(1, 0, “绿巨人”, “冬兵”);

**替换:**

**参数：**

第一个，表示开始位置的索引

第二个，要替换的元素个数

第三个及以后，要替换的新的元素，这些元素将会自动插入到开始位置索引前边

**语法：**

var arr = [2, 3, 4, '绿箭侠', '黑寡妇'];

arr.splice(1, 1, “绿巨人”, “冬兵”); //在下标为1的位置替换成: "绿巨人"然后再插入"冬兵" 因为第二个参数是1 只替换一个元素 剩下的插入

**join() :**

——该方法可以将数组转换为一个字符串

——该方法不会对原数组产生影响，而是将转换后的字符串作为结果返回

——在join()中可以指定一个字符串作为参数，这个字符串将会成为数组中元素的连接符

1.如果不指定连接符,则默认使用" , "作为连接符

2.如果不想用连接符可以传一个""空的字符串

**使用：**

arr.join("+");//用加号连接

**ES3总结**

push()、pop()、shift()、unshift()、sort()、

reverse()、concat()、slice()、splice()、join()

**数组的方法（ES5）：**

**indexOf()**

——该方法可以检索一个数组中是否含有指定内容

——如果数组中含有该内容，则会返回其第一次出现的索引

如果没有找到指定的内容，则返回-1

——可以指定一个第二个参数，指定开始查找的位置

**语法：**

arr2.indexOf(arr[i])

**数组的遍历**

所谓的遍历数组就是把数组中的所有数都取出来

**用for循环遍历**

for(var i=0; i<5; i++){

console.log(arr[i]);

}

**for...in遍历:**

用于遍历数组或者对象

for(var i in arr){

console.log(arr[i]);

}

**forEach()**

—般我们都是使用for循环去遍历数组.

js中还为我们提供了一个方法，用来遍历数组

--这个方法只支持IE8以上的浏览器

IE8及以下的浏览器均不支持该方法，所以如果需要兼容IE8则不要使用forEach，还是使用for循环来遍历数组

**语法：**

需要一个函数作为参数

- 由我们创建但是不由我们调用的，称为回调函数

- 数组中有几个元素函数就会执行几次，每次执行时浏览器会将遍历到的元素以实参的形式传递给这个函数，我们可以定义形参来读取这些内容

- 浏览器会在回调函数中传递三个参数;

第一个参数就是当前被遍历到的元素 value

第二个参数就是被遍历到的元素的索引（下标）

第三个参数就是被遍历的数组