JavaScript Session 2
Array

Carol, SAP
Sep 11, 2017

Definition

Definition in JavaScript

Usage

Create, read and write, length, operate

Array Method

Find, Convert to string, add elements, delete elements, sort

Multi-dimensional Array

Create Bidimensional Array

Iterator

Do not generate new array,
Generate new array

Definition

Standard Definition, Definition in JS

Definition in JavaScript

Store multiple values in a single variable.

var cars = [“Volvo”,”Ford”,”BMW”,”Benz”];

element

var examples = [“Volvo”,2,”BMW”,[1,3,5]];

```
> cars;
[“Volvo”,”Ford”,”BMW”,”Benz”]

> cars.length;
4

> cars[0];
Volvo
```

Usage

Create, read and write, length

Usage Create

```
var empty = [];                                  //没有元素的数组，长度为0的空数组
var primes = [2,3,5,7,11];                       //有5个数值的数组

var base = 1024 ;
var table = [base, base+1, base+2, base+3];      //数组中直接量的值不一定是常量，可以是任意的表达式

var a = [[1,{x:1,y:2}], [2, {x:3, y:4}]];        //其他数组或对象

var b = [1,,3];                                  //数组有3个元素，中间的元素值为undefined
var undefs = [,,];                               //数组有2个元素，都是undefined

console.log(empty.length);                       //显示0
console.log(primes.length);                      //显示5
```

Usage Create

调用时没有参数值

var a = new Array(); //没有元素的空数组

console.log(a.length); //显示0

调用时只传入一个参数，用来指定数组的长度

var a = new Array(10);

console.log(a.length); //显示10

//可以预分配一个数组空间，前提：预先知道所需要元素个数

//该数组中没有存储值，数组的索引属性都没有定义。
//该数组中每个元素的值预定义为undefined

明确指定数组元素的值

var a = new Array(5，4，3，2，1，“testing，testing”);

判断一个对象是否是数组：

var a = 3；
var arr = [1,2,3];
console.log(Array.isArray(a)); // 显示false
console.log (Array.isArray(arr)); //显示true

Usage Read and write

使用[]操作符将数据赋给数组

```
var nums = [];                    //将1-100的数字赋给一个空数组
for (var i= 0;i<100;++i){
    nums[i] = i+1;
}
```

使用[]操作符读取数组中的元素

```
var numbers = [1,2,3,4,5];
var sum = numbers[0]+ numbers[1]+ numbers[2]+ numbers[3]+ numbers[4] ;   //显示15
alert (sum);


var numbers = [1,2,3,5,8,13,21];
var sum = 0 ;
for (var i= 0; i<numbers.length;++i){    //使用for循环来依次读取数组中的所有元素
    sum+= numbers[i];                    //使用length属性来控制循环的次数，可以确保循环遍历了数组中的所有元素
}
alert (sum);                             //显示53
```

Usage length

```
[].length
['a','b','c'].length
```

//=> 0: 数组没有元素

//=> 3: 最大索引为2，length为3

```
a = [1,2,3,4,5];
a.length = 3;
a.length = 0;
a.length = 5;
```

//从5个元素的数组开始

//删除了最后两个元素，此时a=[1,2,3]

//删除所有元素，a = []

//长度为5，但是没有元素，就像new Array（5）

```
var a = [1,2,3];
Object.defineProperty(a,"length",{writable: false});
a.length = 0;
```

//Object.defineProperty()让数组的length属性变成只读

//a不会改变

Usage Create

调用split()方法也可以生成数组

```
var sentence = “the quick brown fox jumped over the lazy dog” ;
var words = sentence.split(“ ”);
for (var i=0;i<words.length;++i){
    alert(“word” + i+ “:”+words[i]);
}
```

```
word 0  : the
word 1  : quick
word 2  : brown
word 3  : fox
word 4  : jumped
……
```

Usage Create

由已有数组创建新数组

```
var cisDept = [“David”, “Cynthia”, “Roy”,”Tom”, “Jennifer”,”Jack”];
var dmpDept = [“Mike”, “Lynn”, “Bryan”];
var itDiv = cisDept.concat(dmpDept);
alert(itDiv);
itDiv = dmpDept.concat(cisDept);
alert(itDiv);
```

执行程序，输出为：
David, Cynthia, Roy, Tom, Jennifer, Jack, Mike,Lynn,Bryan
Mike,Lynn,Bryan, David, Cynthia, Roy, Tom, Jennifer, Jack

```
var itDiv = [“Mike”, “Lynn”, “Bryan”, “David”, “Cynthia”, “Roy”,”Tom”, “Jennifer”,”Jack”];
var dmpDept = itDev.splice(3,3);
var cisDept = itDiv;
alert(dmpDept);
alert(cisDept);
```

位置 个数

执行程序，输出为：
David, Cynthia, Roy
Mike,Lynn,Bryan, Tom, Jennifer, Jack

Usage operation

将一个数组赋给另一个数组

```
var nums = [ ] ;
for (var i=0;i<10;++i){
    nums[i]=i+1;
}
var samenums = nums;
```

```
var nums = [ ] ;
for (var i=0;i<100;++i){
    nums[i]=i+1;
}
var samenums = nums;
nums[0] = 400;
alert(samenums[0]);        //显示400
```

```
function copy(arr1, arr2){
   for(var i=0;i<arr1.length;++i){
       arr2[i] = arr1[i];
   }
}
```

```
var nums = [ ] ;
for (var i=0;i<100;++i){
    nums[i]=i+1;
}
var samenums = [ ];
copy(nums, samenums);
nums[0] = 400;
alert(samenums[0]);        //显示1
```

Array Method

Find, convert to string, add elements, delete elements, sort

find element

indexOf

lastIndexOf()　　//返回位置，例：-1 没有找到相同元素，1，5表示在数组中的[1], [5]位置

```
var names = [“David”, “Cynthia”, “Roy”,”Tom”, “Jennifer”,”Jack”];
prompty(“Enter a name to search for: ”);
var name =readline();
var position = names.indexOf(name);
if (position >= 0) {                                    //在这个数组中至少有一个元素，即判断其是否存在
   alert(“Found”+ name +”at position” + position);
}
else{
  alert(name + “not found in array.”);
}
```

执行程序，输入Cynthia，输出为：
Found Cynthia at position 1

执行程序，输入Lee，输出为：
Lee not found in array.

find element

indexOf

lastIndexOf() //返回位置，例：-1 没有找到相同元素，1，5表示在数组中的[1], [5]位置

```
var names = [“David”, “Cynthia”, “Roy”,”Cynthia”, “Jennifer”,”Jack”];
var name = “Cynthia”;
var firstPosition = names.indexOf(name);
alert(“First found”+ name +”at position” + firstPosition);
var lastPosition = names.indexOf(name);
alert(“Last found”+ name +”at position” + lastPosition);
```

执行程序，输出为：
First found Cynthia at position 1
Last found Cynthia at position 3

Convert to string

join()

toString()

```
var names = [“David”, “Cynthia”, “Roy”,”Tom”, “Jennifer”,”Jack”];
var namestr = names.join();
alert(namestr);
namestr = names.toString();
alert(namestr);
```

执行程序，输出为：
David, Cynthia, Roy, Tom, Jennifer, Jack
David, Cynthia, Roy, Tom, Jennifer, Jack

Add element

push() //将一个元素添加到数组末尾

unshift() //将一个元素添加到数组开头 需要将后面的每一个元素都相应地向后移一个位置

```
var nums = [1,2,3,4,5];
nums.push(6);
alert(nums);      //程序输出 1,2,3,4,5,6
```

```
var nums = [1,2,3,4,5];
nums[nums.length] = 6;   //把[nums.length]设置为6，就是在这个数组nums末尾加一个6
alert(nums);      //程序输出 1,2,3,4,5,6
```

```
var nums = [2,3,4,5];
var newnums = 1;
var N = nums.length;
for (var i=N;i>=0;- - i){   //数组不为空，往前一位
    nums[i] =  nums[i-1];
}
nums[0] = newnums;   //插入
alert(nums);      //程序输出 1,2,3,4,5
```

```
var nums = [2,3,4,5];
var newnums = 1;
nums.unshift(newnums);
alert(nums);      //程序输出 1,2,3,4,5

//通过一次调用，为数组添加多个元素
nums = [3,4,5];
nums.unshift(newnums,1,2);
alert(nums);      //程序输出 1,2,3,4,5
```

Delete element

pop() //删除数组末尾的元素

shift() //删除数组的第一个元素

```
var nums = [1,2,3,4,5,9];
nums.pop();
alert(nums);        //程序输出 1,2,3,4,5
```

```
var nums = [9,1,2,3,4,5];
for(var i=0;i<nums.length;++i){
    nums[i]=nums[i+1];
}
alert(nums);        //程序输出 1,2,3,4,5，
```

需要将前面的每一个元素都相应地向前移一个位置，低效
此外，还多出了一个元素，会多出了一个，

```
var nums = [9,1,2,3,4,5];
nums.shift();
alert(nums);        //程序输出 1,2,3,4,5
```

```
nums = [3,4,5];
nums.unshift(newnums,1,2);
alert(nums);        //程序输出 1,2,3,4,5
```

会将删掉的元素作为方法的返回值返回，
因此可以使用一个变量来保存删除的元素

```
var nums = [6,1,2,3,4,5];
var first = nums.shift();        //first 拿到value 6
nums.push(first);
alert(nums);        //程序输出 1,2,3,4,5,6
```

补充：从数组中间位置添加和删除元素

splice()

为数组添加元素

· 起始索引（希望开始添加元素的地方）
· 需要删除的元素个数（添加元素时该参数设为0）
· 想要添加进数组的元素

```
var nums = [1,2,3,7,8,9];
var newElements = [4,5,6];
nums.splice(3,0,newElements);
alert(nums);        //程序输出 1,2,3,4,5,6,7,8,9
```

不需要var一个新数组

```
var nums = [1,2,3,7,8,9];
nums.splice(3,0,4,5,6);
alert(nums);        //程序输出 1,2,3,4,5,6,7,8,9
```

删除数组元素

```
var nums = [1,2,3,100,200,300,400,4,5];
nums.splice(3,4);
alert(nums);        //程序输出 1,2,3,4,5
```

Sort

reverse() //将数组中元素的顺序进行翻转

sort()

```
var nums = [1,2,3,4,5];
nums.reverse();
alert(nums);                    //5,4,3,2,1
```

如果元素时字符串类型

```
var names =[“David”, “Mike”, “Cynthia”,”Clayton”, “Bryan”,”Raymond”];
names.sort();
alert(names);          //Bryan,Clayton,Cynthia,David,Mike,Raymond
```

如果是数字类型

```
var nums = [3,1,2,100,4,200];
nums.sort();
alert(nums);           //1,100,2,200,3,4
```

传入一个大小比较函数

```
function compare（num1,num2){
    return num1 - num2;
}

var nums = [3,1,2,100,4,200];
nums.sort(compare);
alert(nums);                //1,2,3,4,100,200
```

Multi-dimensional Array

Create Bidimensional array

Bidimensional Array

JS 不支持二维或多维数组，通过元素包含数组的方式，可以间接创建复杂的多维数组。

```
var grades = [[89,99,34],[34,55,70],[1,5,90]];
alert(grades[2][2]);          //显示90
```

处理二位数组的元素，**按列访问**/ 按行访问

```
var grades = [[89,99,77],[89,100,70],[91,88,90]];
var total = 0;
Var average = 0.0;
for(var row=0;row<grades.length;++row){
    for(var col=0;col<grades[row].length;++col){          //因为每一行都是一个数组，可以使用length属性判断每行包含多少列
        total +=grades[row][col];
    }
    average = total/grades[row].length;
    alert("Student" + parseInt(row+1)+"average:"+average.toFixed(2));
    total = 0;
    average = 0.0;
}
```

解析一个字符串，并返回一个整数

把 Number 四舍五入为指定小数位数的数字

程序输出：

Student 1 average:88.33

Student 2 average:86.33

Student 3 average:89.67

Bidimensional Array

处理二位数组的元素，按列访问/ **按行访问**

```
var grades = [[89,99,77],[89,100,70],[91,88,90]];
var total = 0;
Var average = 0.0;
for(var col=0;col<grades.length;++col){
    for(var row=0;row<grades[col].length;++row){
        total +=grades[row][col];
    }
    average = total/grades[col].length;
    alert("Test" + parseInt(col+1)+"average:"+average.toFixed(2));
    total = 0;
    average = 0.0;
}
```

计算一个学生的各科的平均成绩，程序输出：

Test 1 average:89.67

Test 2 average:95.67

Test 3 average:79

Ragged Array

数组中的每行元素个数彼此不同

```
var grades = [[89, 77],[76,82,81],[91,94,89,99]];
var total = 0;
Var average = 0.0;
for(var row=0;row<grades.length;++row){
    for(var col=0;col<grades[row].length;++col){
        total +=grades[row][col];
    }
    average = total/grades[row].length;
    alert("Student" + parseInt(row+1)+"average:"+average.toFixed(2));
    total = 0;
    average = 0.0;
}
```

因为在for循环的内层，计算了每个数组的长度，也就是 数组中每一行的长度不一样

程序输出：

Student 1 average:83

Student 2 average:79.67

Student 3 average:93.25

Iterator

Do not generate new array, Generate new array

Do not generate the array

forEach()

//接受一个函数作为参数，对数组中的每个元素使用函数

```
function square(num){
    console.log(num, num*num);
}

var nums = [1,2,3,4,5,6,7,8,9,10];
nums.forEach(square);
```

执行程序，输出为：

```
1  1
2  4
3  9
4  16
5  25
6  36
7  49
8  64
9  81
10 100
```

every()

//接受一个返回值为布尔类型的函数，对数组中的每个元素使用函数

```
function isEven(num){
    return num % 2 == 0;
}

var nums = [2,4,6,8,10];
var even = nums.every(isEven);
if (even) {
    alert("all numbers are even");
}
else {
    alert("not all numbers are even");
}
```

var nums = [2,4,6,7,8,10];

执行程序，输出为：
not all numbers are even

//当所有元素运用该函数且都返回true，那么这个方法返回true

执行程序，输出为：
all numbers are even

Do not generate the array

some() //接受一个返回值为布尔类型的函数，对数组中的每个元素使用函数

//只要有一个元素应用该函数返回true，那么这个方法返回true

```
function isEven(num){
    return num % 2 == 0;
}

var nums =  [1,2,3,4,5,6,7,8,9,10];
var someEven = nums.some(isEven);
if (someEven) {
    alert("some numbers are even");
}
else {
    alert("no numbers are even");
}
```

执行程序，输出为：
some numbers are even

→ var nums = [1,3,5,7,9];

执行程序，输出为：
no numbers are even

Do not generate the array

reduce() / reduceRight() //接受一个函数，返回一个值

数组中的元素求和

```
function add(runningTotal, currentValue){
    return runningTotal + currentValue;
}

var nums =  [1,2,3,4,5,6,7,8,9,10];
var sum = nums.reduce(add);
alert(sum);                    // 显示55（返回一个值）
```

将数组中的元素连接成一个长的字符串

```
function concat(concatenateString, item){
    return concatenateString + item;
}

var words =  [“the”, “beautiful”,”girl”];
var sentence = words.reduce(concat);
alert(sentence);               // 显示the beautiful girl
```

```
function concat(concatenateString, item){
    return concatenateString + item;
}

var words =  [“the”, “beautiful”,”girl”];
var sentence = words.reduceRight(concat);
alert(sentence);               // 显示girl beautiful the
```

Generate a new array

map() //对数组中的每个元素使用函数,返回一个新的数组

```
function plus(grade){
    return grade += 5;
}

var grades =  [89,90,77,80,92];
var newgrades = grades.map(plus);
alert(newgrades);            // 94,95,82,85,97
```

```
function get(word){
    return word[0];
}

var words =  [“as”, “soon”,”as”,”possile”];
var first = words.map(get);
alert(first.join(“ ”));            // asap

alert(first.toString());      // a, s, a, p
```

Generate a new array

filter()　　//接受一个返回值为布尔类型的函数，对数组中的每个元素使用函数

```
function isEven(num){
    return num % 2 == 0;
}
function isOdd(num){
    return  num % 2 != 0;
}

var nums = [ ];
for(var i=0; i<20;++i){
    num[i]=i+1;
}
var evens = nums.filter(isEven);
alert("Even numbers: ");
alert(evens);
var odds= nums.filter(isOdd);
alert("Odd numbers: ");
alert(odds);
```

//当所有元素应用该函数 且 都返回true，那么这个方法返回一个新的数组

执行程序，输出为：
Even numbers:
2,4,6,8,10,12,14,16,18,20
Odd numbers:
1,3,5,7,9,11,13,14,15,17,19

Thank you.

Contact information:
Carol Shi

Run Simple