

JavaScript进阶

---Array数组

---Date日期

JavaScript进阶

---Array数组



河北师范大学软件学院
Software College of Hebei Normal University

内容提纲

- **数组的创建和基本操作（增删改查）**
- **稀疏数组与多维数组**
- **数组的方法和相关高阶函数**



数组的创建和基本操作

• 创建数组的方式

- 通过字面量的方式直接创建，直接量中的值可以是任意的表达式
- 通过Array构造函数来创建数组对象，注意传递的参数

```
var arr1 = [1,2,3,"4"];  
var arr2 = new Array(5); // 一个参数 思考: new Array("5") 返回什么  
console.log(arr2);  
for(var i=0;i<arr2.length;i++){arr2[i] = i;}  
var arr3 = new Array(1,2,3,4); // 多个参数  
console.log(arr1,arr2,arr3);
```

► (5) [undefined × 5]

► (4) [1, 2, 3, "4"] ► (5) [0, 1, 2, 3, 4] ► (4) [1, 2, 3, 4]

数组的创建和基本操作

• 数组元素的增删改查的基本操作

```
var a = ["hello"];  
a[1] = 3.14; //增: 直接添加数组元素, 通过方法添加元素参见后续章节  
a[2] = "world";  
console.log("删除a[2]前的数组a", a);  
delete a[2]; //删: 思考此时数组长度是2还是3? 如何彻底删除?  
console.log("删除a[2]后的数组a", a);  
a[0] = "XX"; //改: 修改数组元素a[0]  
console.log(a[0]); //查: 看数组中的元素, 索引从0开始
```

```
i = 2;  
a[i] = 3;  
a[i+1] = "YY";  
a[a[i]] = a[0];  
console.log(a); //输出什么?
```



数组的创建和基本操作

• 数组相对于普通对象的特别之处

- 数组是对象的特殊形式，可以为数组添加对象属性，对于0至2的32次方之外的数，将作为普通对象的键来对待
- 数组特别之处在于，当使用使用2的32次方以内的非负整数作为属性名时（包括类型转换的数字），数组会自动维护其length属性，**作为数组的元素，而不是数组对象的属性**

```
var a = [];  
a[-1.23] = true; //创建一个名为“-1,23”的属性  
a["100"] = 0;    // 数组的第101个元素  
a[1.00] = "Hi";  //和a[1]相当  
console.log(a.length);  
console.log(a);
```

101

```
► (101) [undefined × 1, "Hi", undefined × 98, 0, -1.23: true]
```


内容提纲

- 数组的创建和基本操作（增删改查）
- 稀疏数组与多维数组
- 数组的方法和相关高阶函数



稀疏数组与多维数组

• 稀疏数组

- 稀疏数组是包含从0开始的不连续索引的数组，length值大于实际定义的元素个数
- 遍历稀疏数组时，注意的跳过无元素项的问题

```
var a1 = [, "abc"];  
console.log(a1.length);  
for(var i in a1){  
    console.log(i, a1[i]); // 输出几个元素?  
}
```

```
var a2 = new Array(3);  
console.log(a2.length); // 长度为几?  
var a3 = [, ,];  
console.log(a3.length); // 长度为几?
```


稀疏数组与多维数组

• 多维数组（矩形数组、交错数组）

- JS中可以通过包含数组的数组来模拟多维数组

```
var table = new Array(5);  
for(var i=0;i<table.length;i++){  
    table[i] = new Array(5); //若是 table[i] = new Array(i);  
}
```

```
for(var row=0;row<table.length;row++){  
    for(var col=0;col<table[row].length;col++){  
        table[row][col]=row*col;  
    }  
}  
var product = table[2][4];
```

练习：优化代码，for循环合并

内容提纲

- 数组的创建和基本操作（增删改查）
- 稀疏数组与多维数组
- 数组的方法和相关高阶函数



数组的方法

• 数组静态方法（构造器函数对象的方法）

- Array.from(...)、Array.isArray(...)、Array.of(...)等（具体参见实例）

```
function foo(){  
    console.log(Array.isArray(arguments));  
    //console.log(arguments.pop());//这样是否能调用？ 数组与类数组对象  
    console.log(Array.prototype.pop.call(arguments));  
}  
foo(3,2,5);
```

• 数组原型方法（添加和删除元素-破坏性）

- Array.prototype.shift() Array.prototype.unshift(elem1?,elem2?,...)
- Array.prototype.pop() Array.prototype.push(elem1?,elem2?,...)
- Array.prototype.splice(start,deleteCount?,elem1?,elem2?)

数组的原型方法

• 数组原型方法（排序和颠倒元素顺序-破坏性）

- `Array.prototype.reverse()`
- `Array.prototype.sort(compareFunction?)` //回调函数的写法, 思考冒泡排序

• 数组原型方法（合并、切分和连接-非破坏性）

- `Array.prototype.concat(arr1?,arr2?,...)`
- `Array.prototype.slice(begin?,end?)` //注意参数的正负, 注意不要和splice混淆了
- `Array.prototype.join(separator?)` //注意返回的类型

• 数组原型方法（值的查找-非破坏性）

- `Array.prototype.indexOf(searchValue,startIndex?)`
- `Array.prototype.lastIndexOf(searchElement,startIndex?)` //注意方向和起始点

数组相关的高阶函数



• 数组原型方法（迭代-非破坏性-检测方法）

- `Array.prototype.forEach(callback,thisValue?)`
- `Array.prototype.every(callback,thisValue?)` //若有不满足的，立即返回false，不再后续迭代
- `Array.prototype.some(callback,thisValue?)` //若有满足的，立即返回true，不再后续迭代

• 数组原型方法（迭代-非破坏性-转换方法）

- `Array.prototype.map(callback,thisValue?)`
- `Array.prototype.filter(callback,thisValue?)`

• 数组原型方法（迭代-非破坏性-归约方法）

- `Array.prototype.reduce(element,initialValue?)`
- `Array.prototype.reduceRight(callback,initialValue?)`

The background of the slide is decorated with various abstract shapes in shades of green and yellow. These shapes, which include circles, ovals, and teardrop-like forms, are scattered across the top and right sides of the slide, creating a modern and organic feel.

Thank You!



河北师范大学软件学院
Software College of Hebei Normal University

JavaScript进阶

---Date日期



河北师范大学软件学院
Software College of Hebei Normal University

内容提纲

- **Date简介及创建Date对象**
- **Date方法（静态方法、原型方法）**
- **日期和时间格式**



创建Date对象及日期的格式

- Date提供了解析、管理和展示时间的功能
- 通过构造函数创建Date对象的4种形式

`new Date(year,month,date?,hours?,minutes?,seconds?,milliseconds?)` //注意起始索引

`new Date(dateTimeStr)` //参数为字符串类型，注意格式，参见日期格式章节

`new Date(timeValue)` //参数为数字类型，以毫秒为单位

`new Date()` //返回当前世界，相当于`new Date(Date.now())`

```
var date = new Date(2017,9,18,12,34,1);  
console.log(date);
```

Wed Oct 18 2017 12:34:01 GMT+0800 (中国标准时间)

内容提纲

- **Date简介及创建Date对象**
- **Date方法（静态方法、原型方法）**
- **日期和时间格式**



Date的方法

• 静态方法（构造器函数对象的方法）

- Date.now()
- Date.parse(dateTimeString) //转成毫秒，从1970年1月1日 00:00:00开始计算
- Date.UTC(year,month,date?,hours?,minutes?,seconds?,milliseconds?) //转成毫秒 标准时间

• 原型方法

- Date.prototype.get<Unit>() Date.prototype.getUTC<Unit>()
- Date.prototype.set<Unit>() Date.prototype.setUTC<Unit>()
- Date.prototype.getTimeString() Date.prototype.toLocalTimeString()
- Date.prototype.toDateString() Date.prototype.toString() Date.prototype.toLocalString()
- Date.prototype.toISOString() Date.prototype.toJSON()



内容提纲

- **Date简介及创建Date对象**
- **Date方法（静态方法、原型方法）**
- **日期和时间格式**



日期和时间格式

- JS中的时间复合ISO8601扩展格式标准

- 日期格式（无时间）

YYYY-MM-DD、YYYY-MM、YY

- 时间格式（无日期） 注意：考虑宿主环境差异

THH:mm:ss.sss、THH:mm:ss.sssZ、THH:mm:ss、THH:mm:ssZ、THH:mm、THH:mmZ

- 完整的日期时间格式 (T代表时间、Z代表时区信息)

YYYY-MM-DDTHH:mm:ss.sssZ

- 时间比较与运算（转换为毫秒后进行比较和运算）

The background of the slide is decorated with various abstract shapes in shades of green and yellow. These shapes, which include circles, ovals, and irregular blobs, are scattered across the top and right sides of the slide, creating a modern, organic feel.

Thank You!



河北师范大学软件学院
Software College of Hebei Normal University

作业

- 完成freecodecamp任务
- https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Global_Objects/Array