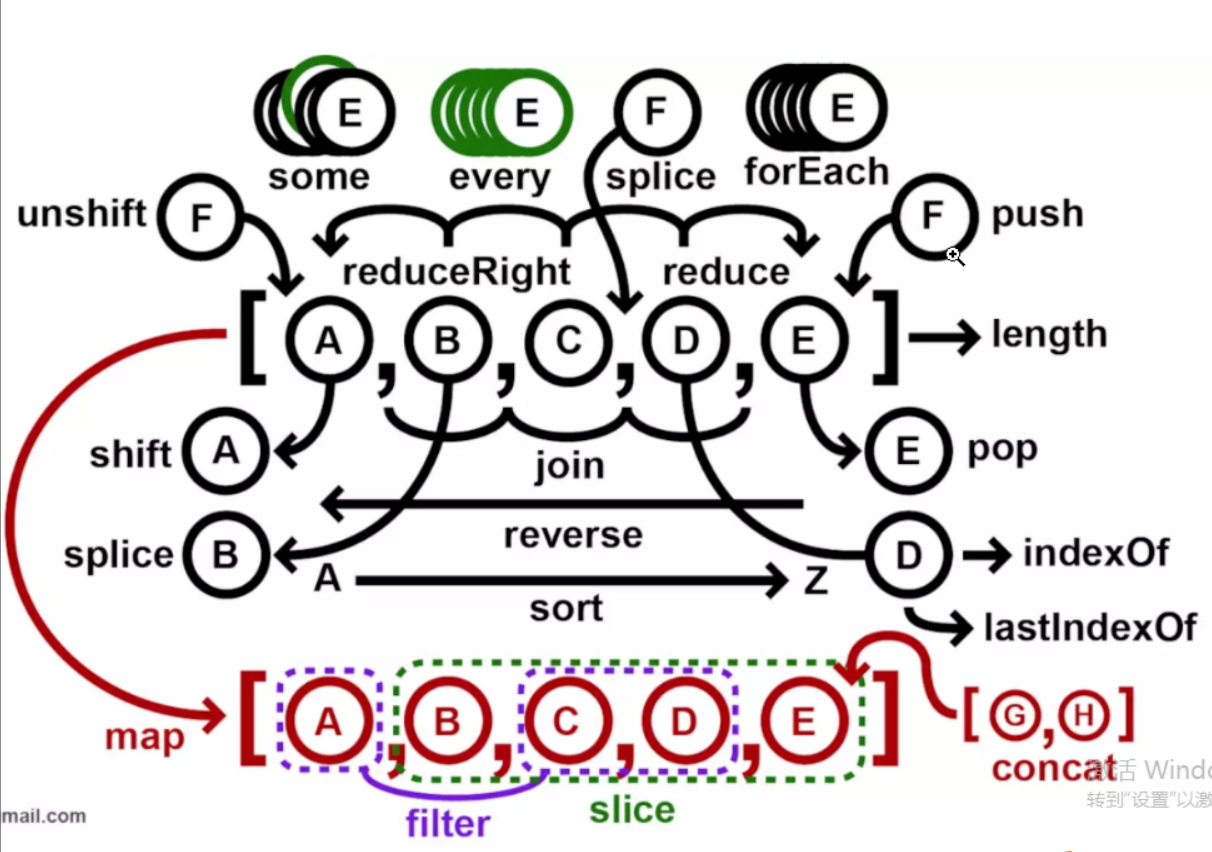
12.16-js10

一、数组的方法



1、isArray()：

1.是Array对象的一个静态方法，用来判断某变量是否是数组类型的变量。返回Boolean。

2.语法：Array.isArray(arr);

3.与typeof不同，typeof只能检测出数组是object类型的。

4.in操作符：用于判断某属性是否存在于某对象或数组中。语法：console.log(6 in arr);

2、every()：

1.检测是否全部符合，一假即假，false则停止执行且every方法直接返回false。此方法的第1个参数是一个回调函数，函数有3个参数。返回Boolean。

2.语法：arr.every(function(item,index,array){代码块;}，第2个参数)。

A、其中item代表每一项，index代表该项的下标（可省略），array代表原数组（可省略）。

B、第2个参数可省略，省略后函数function的this指向window对象。

3、some()：

1.检测是否存在符合，一真即真，true则停止执行且some方法返回true。此方法的第1个参数也是一个回调函数（与every相同）。

2.语法：与every相同。

4、forEach()：

1.Array对象中定义的对数组进行迭代的方法。第1个参数是一个迭代函数（也是回调函数，参数什么的与every中的回调函数相同），第2个是一个this的指向。没有返回值。

2.语法：arr.forEach(function(item,index,array){代码块;}，第2个参数)。

3.特别地：空位元素item不迭代，打印出的empty是由于item指向的index被保留着。

5、keys()：

1.是Object的静态函数，用于遍历对象的可枚举的键名（与getOwnPropertyNames方法不同，get……遍历不管可不可枚举）。返回这些键名所组成的数组。可用于获取对象的长度。

2.语法：Object.keys(arr);

3.特别地：不可迭代空位元素。

6、map()：

1.用于映射数组，可以对每一个元素调用回调函数（回调函数可以是系统的内置函数）。返回一个 包含每一个回调函数的返回值 的数组。

2.语法：同于every()。

3.特别地：不会空位元素进行迭代。

4.面试题的坑：var re = [1,2,3].map(parseInt);console.log(re); //https://blog.csdn.net/willard\_cui/article/details/81504782

7、filter()：

1.用于筛选出数组中满足条件的元素并组成新的数组返回。如果回调函数返回true则表明当前元素满足条件，反之则不满足。

2.语法同于every

3.特别地：不对空位元素迭代。

8、reduce()：

1.用于累加，可以对每一个元素调用回调函数（该函数返回的是累计结果，该结果用于下一次调用回调函数）。

2.语法：arr.reduce(a为回调函数上次返回的值<函数没有返回值则为undefined，没有第2个参数指定a为什么东西的话，则默认a为下标为0的元素>，b为本次进入函数的值，c为下标，d为数组)

9、练习

10、总结：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **作用** | **返回** | **语法** | **特别地** | **例** |
| **every()：** | **检测是否全部符合，一假即假，false则停止执行且返回false。** | **Boolean** | **arr.every(function(item,index,array){代码块;}，第2个参数)；** | **第2个参数可省略，省略后函数function的this指向window对象。** |  |
| **some()：** | **检测是否存在符合，一真即真，true则停止执行且返回true。** | **Boolea** | **同上** | **同上** |  |
| **forEach()：** | **数组迭代** | **无** | **同上** | **不会迭代空位元素。** |  |
| **map()：** | **映射数组** | **返回一个包含回调函数的返回值的数组** | **同上** | **不会迭代空位元素。** |  |
| **filter()：** | **筛选出数组中满足条件的元素** | **返回满足条件的元素组成的新数组** | **同上** | **不会迭代空位元素。** | **过滤出2的倍数：**  **var newArr = arr.filter(function (item) {**  **return item%2==0;**  **})** |
| **reduce()：** | **用于累加** | **返回一个与原数组长度一样的新数组** | **arr.reduce(a为回调函数上次返回的值，b为本次进入函数的值，c为下标，d为数组)；** | **<回调函数没有返回值则为undefined，没有第2个参数指定a为什么的话，则默认a为下标为0的元素>** |  |

二、迭代数组for与for in的区别

1、for的迭代需要使用数组的长度 执行效率比for/in要低

**var *arr*** = [**"a"**,**"b"**,**"c"**,,**"e"**];  
**for** (**var *i*** = 0; ***i*** < ***arr***.**length**; ***i***++) {  
 ***console***.log(***arr***[***i***]);*//a b c undefined e*}

2、for in 是跳过数组的空位的

**for**(**var *j* in *arr***){  
 ***console***.log(***j***,***arr***[***j***]);*//a b c e*}

三、计时器

1、setTimeOut()：

1.语法：window.setTimeOut(code,delay)。

2.用于在指定的时间之后去执行相应的代码。

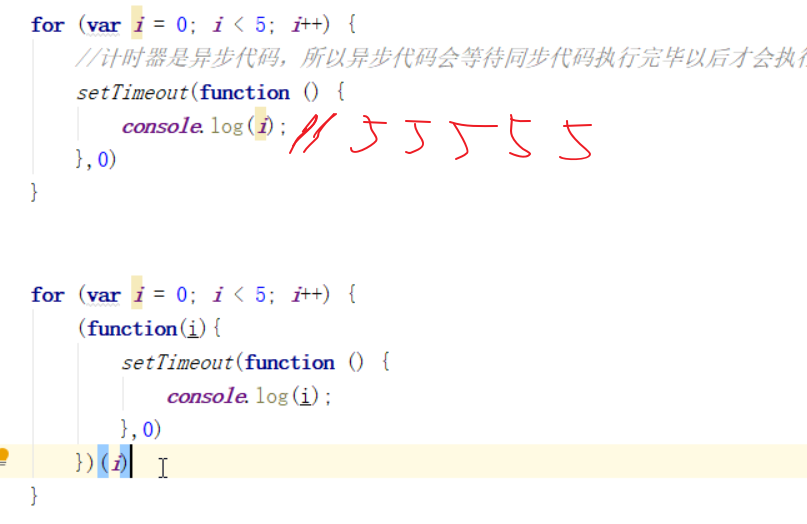
3.参数code：表示要延迟执行的代码（将代码放在window的环境中执行），可以是字符串（默认eval方法执行了代码），也可以是函数（一般建议设置一个函数，如果这个函数需要传参，那就设置成setTimeOut方法的第3个参数）。

4.参数delay：设置延迟时间，以毫秒为单位（1s = 1000ms）。

5.第3个及以后的参数都作为计时器函数的实参。

6.返回值是一个timerID，这个ID是唯一的，指向延迟执行代码的句柄。这个ID在清除计时器时可以使用。

7.计时器遵循defer类型的异步代码执行原则。



2、clearTimeOut()：

1.语法同setTimeOut()。

2.clearTimeOut(timerX);

3、setInterval()：

1.间歇计时器。

2.参数code：同于setTimeOut()的code。

3.参数interval：代表间歇时间，刷新页面后先延迟再执行。

4、clearInterval()：



5、计时器面试题：

