**什么是 AMP? <https://amp.dev/zh_cn/>**

AMP 的全称是 Accelerated Mobile Pages。 是一种为快速展示静态内容而制作网页的计划。



快速了解 AMP (Accelerated Mobile Pages)

**<https://blog.csdn.net/kinsxkins/article/details/80494655>**

**ES6中setTimeout函数的执行上下文**

<https://blog.csdn.net/liwusen/article/details/56278944>

简介：setTimeout，延迟执行函数里的执行上下文

（1）ES5中，setTimeout里面的函数的执行上下文为全局上下文，举例来说：

function log(){

setTimeout(function(){console.log(this.id)},100)

}

var id=42

log.call({id:21})

//输出的结果为42

我们可以看到setTimeout，被延迟执行的函数，里面的this，指向的是全局作用域，也就是这个函数的上下文为全局上下文。

（2）在ES6的箭头函数中，setTimeout里面，如果执行了一个箭头函数，那么这个函数的执行上下文为定义这个箭头函数所在的函数。

function log(){

setTimeout(()=>{

console.log(this.id)

},100)

}

var id=42;

log.call({id:21})

//这次输出的是21

总结：

因为ES6中，箭头函数的this，规定的指向定义这个箭头函数所在的那个函数。于是这里箭头函数里面的this,就固定bind了{id:21}。

**前端面试题 - 如何将 Set 集合转化为一个数组**

麦田里的POLO桔 于 2021-05-08 23:20:41 发布

什么是 Set 集合：

Set 集合类似于一个数组，但是它要求里面是元素不可以有重复。

const set = new Set(['welcome', 'you', '!', 'you']);

console.log(set); // Set { 'welcome', 'you', '!' }

可以看到，Set 的实例是一个对象，这个对象的内部结构很像一个数组，传递给 Set 构造函数的数组，其内部重复的元素最终只会保留一个。

将 Set 集合转化为一个数组的方法：

1. 使用 Array.from() 方法

let set = new Set([12, 45, 97, 9797, 564, 134, 45642, 12, 564])

console.log(set) // Set { 12, 45, 97, 9797, 564, 134, 45642 }

console.log(Array.from(set)) // [ 12, 45, 97, 9797, 564, 134, 45642 ]

简单地介绍一下 Array.from() 这个函数：

Array.from() 方法就是将一个类数组对象或者可遍历对象转换成一个真正的数组。

那么什么是类数组对象呢？所谓类数组对象，要具备两个要求：

最基本的要求就是具有 length 属性，

其次就是属性名要是非负整数或者字符串类型的非负整数。

虽然它表现出了数组的一些性质，但是该对象并不是由 Array 构造函数所创建的。

Array.from(obj, mapFun, thisArg) 有三个参数：

obj：被转化为数组的对象；

mapFun：作用类似于数组的 map 方法，可以对每个元素进行处理，将处理后的值放入返回的数组；

thisArg：this关键字的指向（通常都不用它）。

var arr = [12, 45, 97, 9797, 564, 134, 45642]

var set = new Set(arr)

var resultArr = Array.from(set, function(item){

return (item + 1);

})

console.log(resultArr) // [ 13, 46, 98, 9798, 565, 135, 45643 ]

传递给 Array.length 的类数组对象属性名可以不连续，也不一定非要从小到大，其次还可以重复。

最终，Array.from() 会将这些属性按照从小到大的顺序存放在数组对应位置中，下标没有出现过的位置将被赋予 undefined，属性名重复时，后一个属性值会覆盖前一个属性值。

一定要牢记：length 一定要大于所定义的最大属性名，而不能仅仅等于属性个数，否则属性名超过 lenth-1 的属性值将不会被添加到数组中去。

var arrayLike = {

3: 'tom',

1: '65',

3: '男',

"4": ['jane','john','Mary'],

'length': 5

}

var arr = Array.from(arrayLike)

console.log(arr) // [ undefined, '65', undefined, '男', [ 'jane', 'john', 'Mary' ] ]

其次也可以传递一个字符串（可遍历对象，Set 实例对象也是一个可遍历对象）：

var str = 'hello world!';

console.log(Array.from(str)) // ["h", "e", "l", "l", "o", " ", "w", "o", "r", "l", "d", "!"]

2. 使用三点运算符 " ... "

const set = new Set(['HELLO', 'JS']);

console.log(set); // Set { 'HELLO', 'JS' }

const array = [...set];

console.log(array); // [ 'HELLO', 'JS' ]

简单的介绍一下三点运算符"..."

在 ES6 规范中三点运算符有两个用途

参数使用，

拓展运算符使用。

参数使用

作为参数使用的三点运算符，也可称为剩余参数，就是将剩余的所有参数放至一个数组内；

剩余参数传入的参数可不限数量；

剩余参数只能出现在最末尾，否则报错。（当然以扩展运算的身份出现时另当别论）

(function foo(param1, param2, ...params){

console.log(param1 + ", " + param2 + ", " + params + ", " + Object.prototype.toString.call(params));

//a, b, 1,2,3,456, [object Array]

})("a", "b", 1, 2, 3, "456");

拓展运算符使用

返回数组中的各项数组拆解

返回数组中的各项，作为扩展运算符身份使用时，可以放在中间

var a = [2, 3];

console.log(1, ...a, 4); //1 2 3 4

返回字符串的每个字符：

var str = "abcde";

arr = [...str]; // 分解字符串，并放入数组

console.log(arr); // ["a", "b", "c", "d", "e"]

3. 使用 forEach

var set = new Set();

set.add("Hello").add("World").add("!");

console.log(set); // Set { 'Hello', 'World', '!' }

var Array = [];

set.forEach(function(item, index, arr) {

Array.push(item);

});

console.log(Array); // [ 'Hello', 'World', '!' ]

————————————————

版权声明：本文为CSDN博主「麦田里的POLO桔」的原创文章，遵循CC 4.0 BY-SA版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/qq\_44647809/article/details/116540100