前端面经复习20220808

1. JS中==和===的区别：

==：宽松相等； ===：严格相等； ==允许在相等的比较重进行强制类型转换，===不允许

2.数组去重

<1>.利用ES6 Set去重（ES6中最常用）

```function 复制代码
        //return Array.from(new Set(arr))
    }
    var arr = [1,1,'true', 'true',true,true,15,15,false,false,undefined,undefined,
null,null,NaN,NaN,'NaN',0,0,'a','a',{},{}];
console.log(unique(arr))
```

但这种方法无法去掉“{}”空对象。

<2>.利用for嵌套for，然后splice去重（ES5中最常用）

```function 复制代码
        for(var i=0;i<arr.length;i++){
           for(var j=i+1;j<arr.length-1;j++){
               if(arr[i]==arr[j]){
                   arr.splice(j,1)
                   j--
               }
           }
        }
        return arr
    }
    var arr = [1,1,'true', 'true',true,true,15,15,false,false,undefined,undefined,
null,null,NaN,NaN,'NaN',0,0,'a','a',{},{}];
console.log(unique(arr))
```

3.JS中几种内存泄露的情况

1. 意外的全局变量
2. 闭包
3. 未被清空的定时器
4. 未被销毁的事件监听
5. DOM引用

4.JS异步任务

JS的异步任务分为宏任务和微任务两种；

1. 宏任务

setTimeout、setInterval、setImmediate、postMessage、MessageChannel等（仅需知道前三种）

2.微任务

Promise、async、await、generator等

3.实现异步的方法

回调函数、事件监听、发布订阅、Promise/A+、生成器Generators/yield、async/await

5.怎么解决回调地狱

1. Promise
2. Async/await

6.call apply bind的作用和区别

作用：都可以改变函数内部的this指向； 区别：1.call和apply会调用函数，并且改变函数内部的this指向 2.call和apply传递的参数不一样，call传递参数arg1，arg2...形式apply必须数组形

式[arg] 3.bind不会调用函数，可以改变函数内部this指向

！！！箭头函数无法使用call apply bind改变this指向。因为其this值在函数定义的时候就已经被定义下来了。

7.箭头函数与普通函数的区别

1. 语法更加简洁、清晰；
2. 箭头函数不会创建自己的this；它会捕获自己在定义时所处的外层执行环境的this。
3. 箭头函数继承而来的this指向永远不变；
4. call()、apply()、bing()无法改变箭头函数中的this指向。
5. 箭头函数不能作为构造函数使用
6. 箭头函数没有自己的arguments
7. 箭头函数没有原型prototype

8.ES6新特性

1. let、const、块级作用域和变量声明；

let声明的变量仅在所在块中生效；不存在变量提升。const变量行为与let类似，只是多了两点更强的约束：1.声明时必须复制；2..声明的变量内存地址不可变。

2.原生对象的方法扩展

- 2.1**String** 加强了对unicode的支持、支持字符串遍历、repeat()等方法的支持、模板字符串；
- 2.2**RegExp** 构造函数第一个参数是正则表达式，指定第二个参数不再报错、u修饰符、y修饰符、s修饰符
- 2.3**Number** 二进制和八进制新写法、新方法parsInt()、Number.EPSILON极小常量、安全整数、Math新方法
- 2.4**Function** 函数参数默认值、rest参数、函数内部严格模式、函数的name属性、箭头函数
- 2.5**Array** 扩展运算符... 形式为（...变量名），用于获取函数的多余参数，这样就不需要使用arguments对象了。
- 2.6**Object和Symbol** （待补充）