前端面试题

简述一下对节流和防抖的理解。

* 防抖
* 原理：在事件被触发n秒后再执行回调，如果在这n秒内又被触发，则重新计时。
* 适用场景:
  + 按钮提交场景：防止多次提交按钮，只执行最后提交的一次
  + 搜索框联想场景：防止联想发送请求，只发送最后一次输入
* 实现：

const debounce = (fun, delay = 500) => {  
let timer = null //设定一个定时器  
return function (...args) {  
clearTimeout(timer);  
timer = setTimeout(() => {  
fun.apply(this, args)  
}, delay)  
}  
}

* 节流
* 原理：- 规定在一个单位时间内，只能触发一次函数。如果这个单位时间内触发多次函数，只有一次生效。
* 适用场景
* 拖拽场景：固定时间内只执行一次，防止超高频次触发位置变动
* 缩放场景：监控浏览器resize
* 实现：

const throttle = (fun, delay = 1000) => {  
let flag = true;  
return function (...args) {  
if (!flag) return;  
flag = false  
setTimeout(() => {  
fun.apply(this, args)  
flag = true  
}, delay)  
}  
}

简述一下对代码规范的理解，并举例几条JS的基本规范。

* > 开放题，言之有理即可。
* > 如：
* > 好处：
* > **1、从根本上降低开发成本：**
* > 提高代码整体的可读性、可维护性、可复用性。
* > **2、保证代码的一致性：**
* > 软件系统中最重要的因素之一就是编码的一致性。如果编码风格一致，也更加易于维护，因为团队内任何人都可以快速理解并修改。
* > **3、提升团队整体效率：**
* > 开发人员通常需要花费大量的时间来解决代码质量问题，如果都按照规范编写，也有助于团队尽早发现问题，这将提高整个交付过程的效率。 > 弊端： > **1、增加团队成员间的协作负担：** > 由于缺乏规范，导致代码风格不一，极端情况下，某段代码只有某个人能修改。 > **2、团队间协作更加困难：** > 由于开发人员要适应不同的风格，会导致效率低下。 > **3、回顾困难：** > 在review期间，可能经常为类似的事情做过多的讨论。 > **4、影响降低团队整体效率：** > 影响团队的生产力和质量，严重的甚至会影响团队和谐。
* > 代码规范例子： > 1、不要在同一行声明多个变量 2、请使用===/！==来比较true/false或者数值 3、使用对象字面量替代new Array这种形式 4、不要使用全局变量 5、switch语句必须带有default分支 6、函数不应该有时候有返回值，有时候没有返回值 7、for循环必须使用大括号 8、if语句必须使用大括号 9、for-in循环中的变量 应该使用let关键字明确限定作用域，从而避免作用域污染

vue3.x中怎么监听对个数据的变化？

* >
* - ref watch([sum, tips], (newVal, oldValue) => { console.log(newVal, oldValue);// [121, ‘a’]，[78, ‘b’]监听结果新的值和旧的值都是数组 });
* reactive watch([() => obj.test, () => obj.year], (newVal, oldValue) => { console.log(newVal, oldValue); //得到的是对象 } );
* 分单个watch进行监听

请简单列举几种水平垂直居中的CSS写法。

* > - absolute + margin 负值
* > - absolute + margin auto
* > - absolute + translate
* > - Flex
* > - table + table-cell + vertical-align: middle