**javascript笔记(梁益飞)**

**javascript中推荐使用let 而不是var**

**还有const 也是块作用域，但是它是不可变的常量，这意味着一旦变量被定义并赋值，就不能改变其值**

**因为，let和var有一些重要的区别，主要的区别在于作用域，使用var声明的变量是函数级别作用域，而使用let声明的变量是块级别作用域**

**块级别作用域 就是一对花括号内的代码，如果你尝试在代码块之外访问这些变量，将会报错。**

**使用块级别作用域，可以避免因变量提升和全局命名冲突导致的意外错误。此外，当代码块执行完毕后，在块中定义的变量也会被自动销毁，从而释放内存空间。****如果你尝试在代码块之外访问这些变量，将会报错。**

**简单点表示**

{

    // 这里是一个块级作用域

    let x = 10;

    var y = 20;

}

console.log(x); // 输出：ReferenceError: x is not defined

console.log(y); // 输出：20

**在上面的代码中，因为 x 是使用 let 声明的，所以它只在块级作用域内可见，不能在块级作用域之外访问。**

**另一方面，由于 y 是使用 var 声明的，它属于函数作用域或全局作用域，并且可以在块级作用域之外访问。**

**所以var不推荐**

        function example() {

  console.log(x); // 输出undefined

  var x = 1;

}

**Var还有变量提升问题**

**由于 var x = 1; 在输出语句的后面，没有输出我们赋值的1**

        function example() {

  console.log(x); // 输出1

  let x = 1;

}

**而let就不会发生这种问题，因为let是块作用域**

**json文件是用来存储数据的，而js文件则是包含可执行代码的**

**JSON最初由Douglas Crockford在2001年提出，现已成为一种流行的数据交换格式。它已经被广泛应用于Web应用程序中，是API返回数据的常用格式，也常用于存储和传输结构化数据。**

**在前端开发中，JSON通常用于与后端进行数据通信，或者作为本地存储格式。**

**Vscode自动生成的settings.json文件**

{

    "liveServer.settings.port": 5501

}

/\*

    "liveServer.settings.port": 5501 是一个 VS Code 编辑器的配置设置，它指定了 Live Server 插件的端口号。

    Live Server 是一个可以在本地开发环境中提供实时服务器和网页预览功能的插件。

    通常情况下，当你调用 Live Server 插件时，默认会在自动帮你分配一个可用的端口号来运行实时服务器。

    但是在某些情况下，你可能想要手动指定端口号以满足特定需求。在这里，"liveServer.settings.port" 就起到了这个作用。其中 5501 是指定使用的端口号。

    请注意，并非所有配置文件都包含此设置。如果你在你的 JSON 配置文件中没有看到此设置，则默认为使用自动分配的端口。

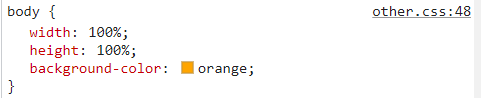
\*/

**有的json配置文件是可以注释的，但是json标准文件是不能注释的**

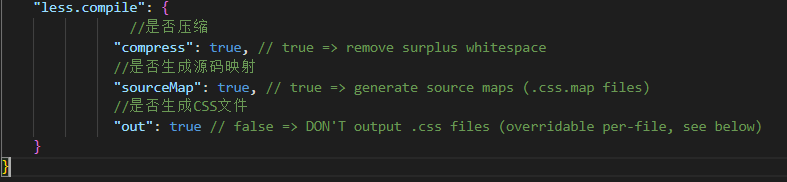
**配置 Easy less文件中的 settings.json**

**为什么要设置 settings.json ，因为，在浏览器中看到的代码列数是css的文件，但是我们改动代码是需要去less文件中，更改的文件在自动生成为css**

**没有更改settings.json文件时，google chrome 会给出css代码的位置**

****

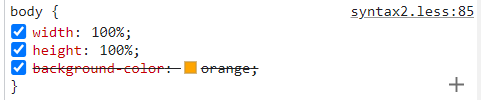
**更改settings.json文件**

****

**自动生成的json标准文件**

**sp20230609_222308_419**

**更改settings.json文件后，google chrome 会给出less代码的位置**

****

**API是javascript中的重要的技术**

**API就是应用程序接口 API是已经建立好的一套代码工具**

**为什么有些网站你进入，它会调用你的地理位置，这就是因为他的网站里面调用了地理位置API ,而这些API就是别人写好的,你只需要会调用和使用**

**还有第三方API比如调用twitterAPI 和 新浪微博的API**

**浏览器安全**

**每个浏览器标签页都是用来运行代码的独立容器,大多数情况下每个标签页中的代码完全独立运行,而且一个标签页中的代码不能直接影响另一个标签页,如果不这样,黑客就可以从其他网站盗取信息**

Javascript中的**querySelectorAll()**方法

querySelectorAll()是javascript中的一个方法,用于从文档中选择或者查找与指定选择器组匹配的元素,并返回一个NodeList 对象.其中包含了这些匹配的元素

NodeList 是一个类似数组的对象,它包含了匹配的元素,可以通过索引访问它们

该方法接受一个参数,即选择器字符串,用于指定要选择的元素,

选择器可以是css选择器 用于选择匹配指定条件的元素 比如:

var elements = document.querySelectorAll(‘.class-name’);

上述代码会选择文档中所有具有”class-name”类名的元素,并将它们存储在一个NodeList对象

需要注意哦 querySelectorAll()方法返回的是一个静态的 NodeList对象 即使文档中的元素发生变化,也不会自动更新返回的结果

如果需要动态的选择元素,可以使用类似的方法如

**getElementsByTagName()**

**getElementsByClassName()**

**querySelectorAll()**方法有什么用

querySelectorAll()方法在javascript中非常有用,它可以实现以下功能

1.选择符合特定条件的元素:可以使用css选择器语法来选择文档中的元素

例如: 可以选择所有具有特定类名,具有特定属性,带有特定标签名等等的 元素

1. 遍历匹配的元素: 返回NodeList 对象包含了匹配的元素,可以通过循环迭代遍历这些元素,并对它们进行操作,例如: 可以修改元素的样式,添加/删除元素的类名,添加事件处理程序等
2. 动态更新选择的元素: 由于querySelectorAll()返回的是一个静态的NodeList对象,可以将结果存储在变量中,以便之后使用或在不同的操作中共享

即使,文档中的元素发生变化,返回结果也不会随着自动更新,这样的特性在需要缓存选择结果的情况下非常有用

总而言之 querySelectorAll()方法 提供了一种灵活和方便的方式来选择和操作文档中的元素

<script>标签中,**defer**有什么作用?

1. **延迟脚本的加载:当浏览器遇到带有defer属性的<script>标签时,会继续解析HTML文档,而不必立即下载和执行脚本文件。这样可以提高页面的加载速度，特别是当脚本文件较大或需要较长的加载时间时**
2. **延迟脚本的执行：一般情况下，带有 defer 属性的脚本会在整个文档解析完毕之后、DOMContentLoaded 事件触发之前执行。这意味着脚本会在文档完全加载和解析完成后再执行，而不会阻塞文档的解析和渲染过程。**

**在网站中遇到的问题之mkv视频在另一个标签页播放问题**

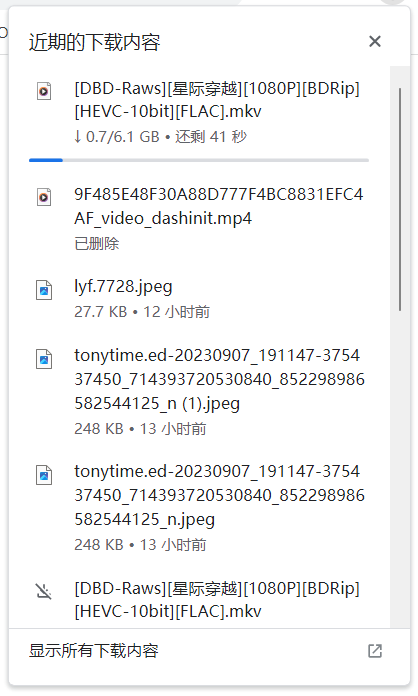
**sp20231004_105126_954**

**这是MP4视频在a标签中的效果,可以看到点击a连接,电影播放没有问题**

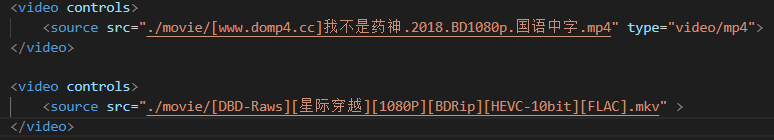
****

**sp20231004_105118_952**

**这是mkv视频的在a标签的效果,点击之后开始了下载,电影并未播放**

****

**使用video标签解决mkv视频的在线播放**

****



**这样的确可以解决mkv电影无法播放的问题,**

**可以是我们需要点击时电影会在新的页面上播放.**

**a标签可以对MP4完美，既不会下载MP4，而且在新的页面播放**

**使用JS解决MKV电影无法在video中打开新标签页**

**设置一个按钮用于播放mkv视频**

**sp20231004_112559_589**

**JS代码**

****

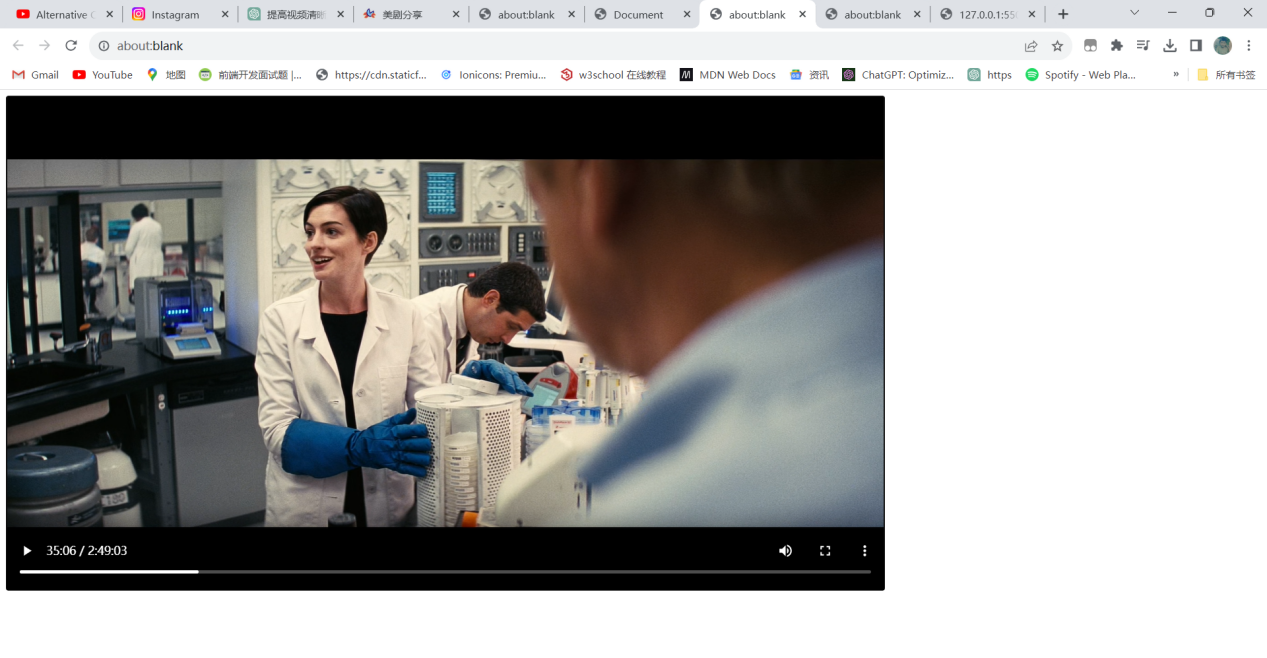
**点击播放按钮**

**sp20231004_112831_003**

**JS效果 非常完美!!**

**在另一个网页中打开电影，不会影响主页面，**

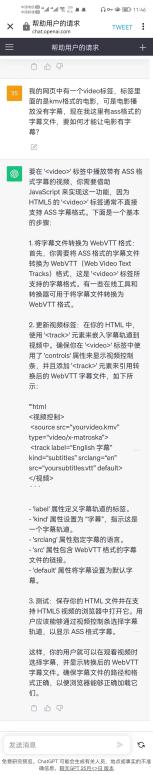
**而且可以播放了，不会下载影片了,这就是我要的效果，但是.........！！**

****

**解决KMV电影的字幕问题**

**没错这个播放问题虽然解决了,但是字幕没有!! 需要将字幕文件导入到电影里!!**

**累死我了！！不过还好有chatGPT！！**

****

**可以点击放大镜查看**

**这里有两种办法 1.将ass字幕转换成webVTT字幕 2.使用JS**

**第二种方法需要使用到ffmep.wasm这是一个开源的多媒体处理工具**

**还需要使用到hls.js这是一个js库**

**由于 还不会使用，所以下次在尝试，我选择第一种方法**

**微信图片_20231004181048**

这是第二种方法图片,可以点击放大镜查看

**这是第二种方法的js代码**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>视频播放器</title>**

**</head>**

**<body>**

**<video id="videoPlayer" controls style="max-width: 70%;"></video>**

**<script src="ffmpeg.wasm.js"></script>**

**<script src="hls.js"></script>**

**<script>**

**// 创建 video 元素**

**const videoElement = document.getElementById("videoPlayer");**

**// 异步加载 ffmpeg.wasm**

**const ffmpeg = createFFmpeg({**

**log: true,**

**});**

**(async () => {**

**await ffmpeg.load();**

**// 读取 KMV 视频文件**

**const kmvFile = "your\_video.kmv";**

**await ffmpeg.FS("writeFile", "input.kmv", new Uint8Array(await fetch(kmvFile).then(response => response.arrayBuffer())));**

**// 读取 ASS 字幕文件**

**const assFile = "your\_subtitles.ass";**

**await ffmpeg.FS("writeFile", "subtitles.ass", new TextEncoder().encode(await fetch(assFile).then(response => response.text())));**

**// 使用 FFmpeg 将字幕嵌入到视频中**

**await ffmpeg.run("-i", "input.kmv", "-vf", "ass=subtitles.ass", "output.mp4");**

**// 读取合并后的视频**

**const outputData = await ffmpeg.FS("readFile", "output.mp4");**

**// 创建 Blob 对象用于视频源**

**const blob = new Blob([outputData.buffer], { type: "video/mp4" });**

**// 创建视频源并添加到 video 元素**

**const source = document.createElement("source");**

**source.src = URL.createObjectURL(blob);**

**videoElement.appendChild(source);**

**// 使用 hls.js 播放视频**

**const hls = new Hls();**

**hls.loadSource(URL.createObjectURL(blob));**

**hls.attachMedia(videoElement);**

**hls.on(Hls.Events.MANIFEST\_PARSED, () => {**

**videoElement.play();**

**});**

**})();**

**</script>**

**</body>**

**</html>**

**第二种方法**

**<https://subtitletools.com/convert-to-vtt-online>**

**点击这个网站 将ass字幕文件转换成vtt**

**输入这行代码**

<track label="English Subtitles" kind="subtitles" srclang="zh-CN" src="./movie/subtitle/星际穿越zh-CN.vtt" default>

**效果:**

****

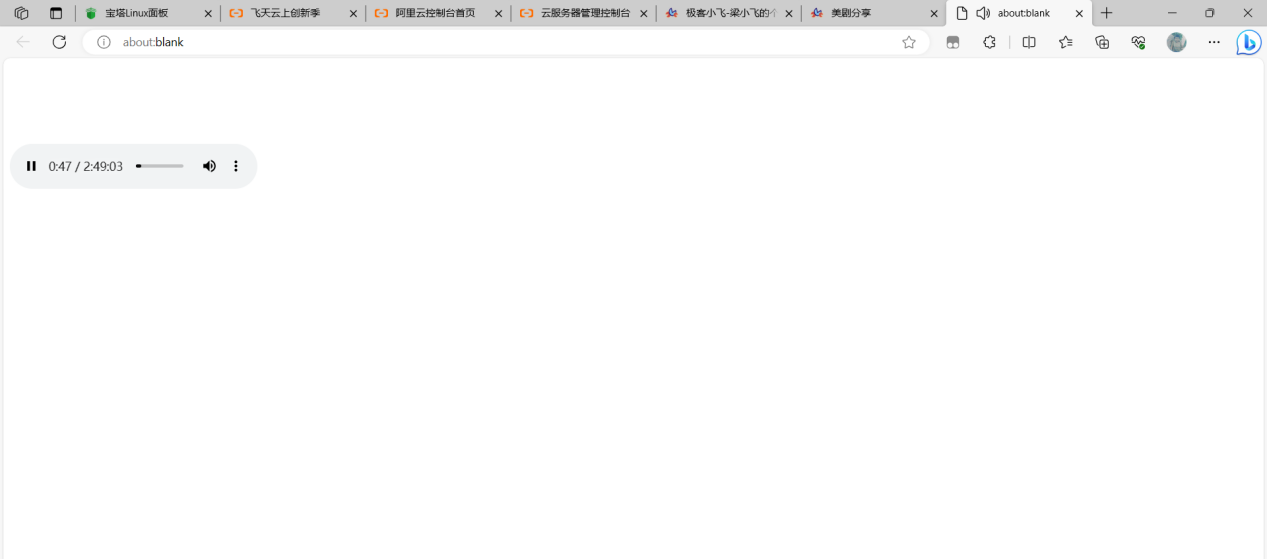
**终于是将这些问题解决了！！**

**对<track>标签的一些解释**

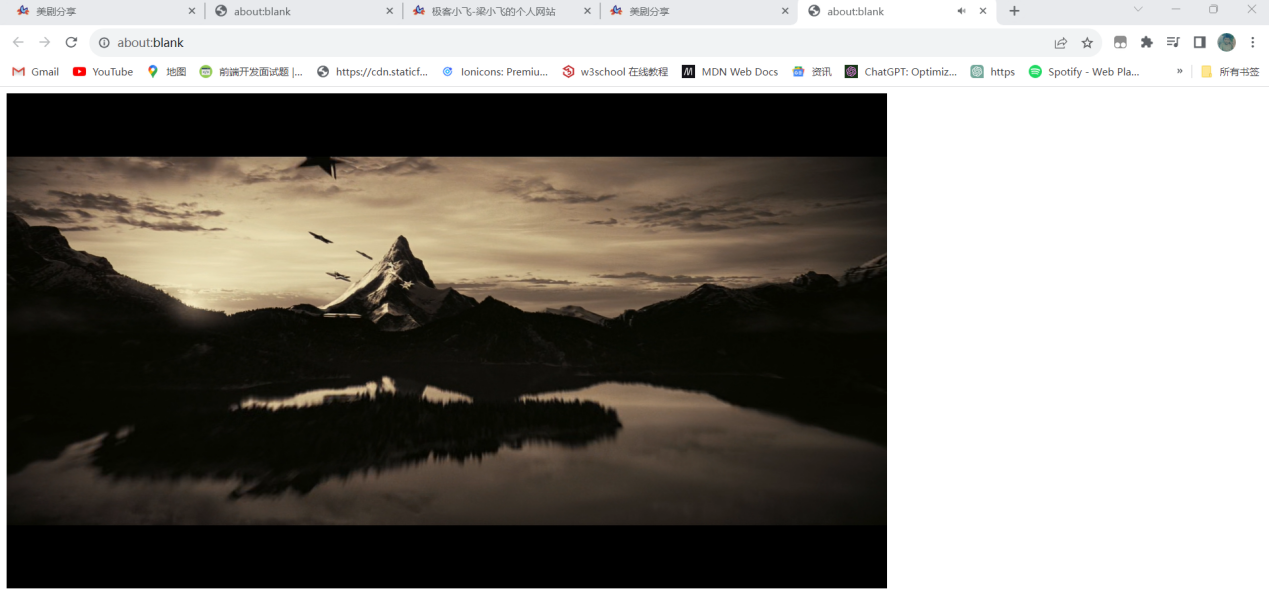
****

**关于《星际穿越》在线观看影片上线后的无法观看问题，只有字幕和声音!!没有画面!!**

**妈的!!又有问题了!!在本地上这个《星际穿越》完全没有问题，但是!!一到服务器中**

**电影完全放不出来!!只有声音！！**

**又需要将mkv格式转换成MP4格式!!!**

**问题妈的很神奇!!自己恢复了!!我过了一晚上来又来访问这个mkv格式的《星际穿越》可以访问了!!!太神奇了!!!**

**妈的问题又来了!点击后又没有电影，搞了这么久，最后的解决办法是，将mkv格式转换为MP4..........**

**介绍innerHTML属性**

**什么是属性, 而什么是方法?**

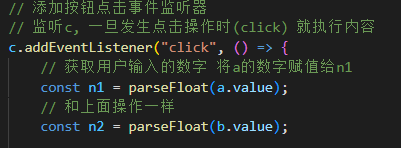
**属性:**

<!-- 属性（Property）：属性是对象的特征或状态，通常用于获取或设置对象的某些值。d.innerHTML 是一个属性，它用于获取或设置 HTML 元素的文本内容。 -->

**方法:**

 <!-- 方法（Method）：方法是对象的行为或功能，通常用于执行某些操作。方法是一个函数，可以在对象上调用以执行特定的任务。例如，addEventListener 是一个方法，用于在元素上添加事件监听器。 -->

**解释一下 addEventListener 这个方法**

****

**继续解释 innerHTML**

**innerHTML这个属性，它用于获取或设置HTML元素的内部HTML内容**

**这个属性允许您读取元素内部的 HTML 代码，也可以用于替换元素的内部内容。**

**sp20231008_222024_165**

**利用JS就可以修改HTML中的内容**

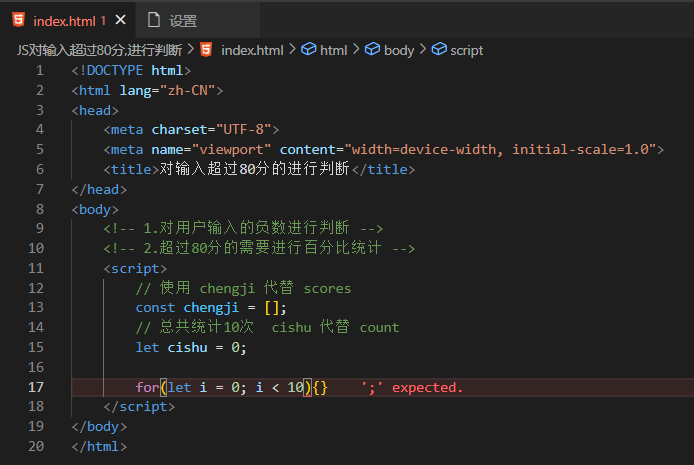
**解释 const n1 = parseFloat(a.value);**

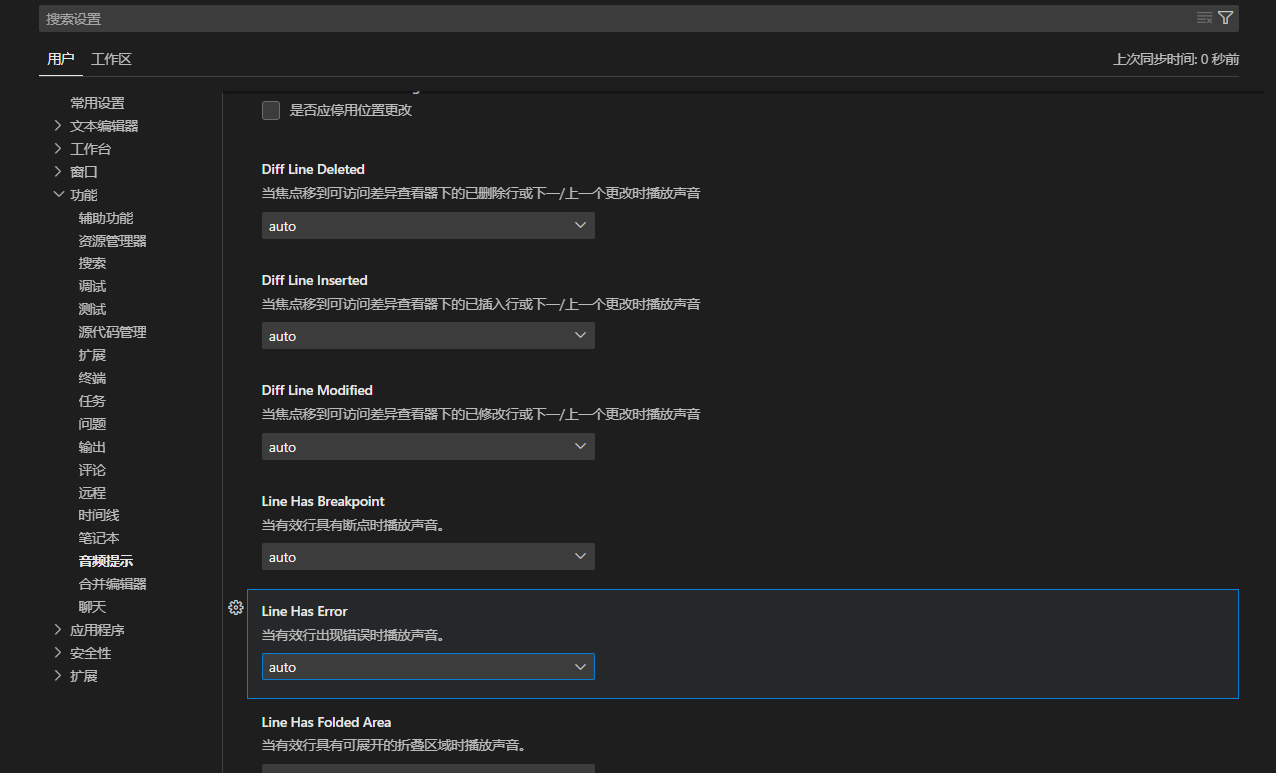
**获取用户在id=“a”中的输入的数字，并把数字赋值给n1**

**a.value 用于获取输入框中的内容 这里的value是输入框的属性 用于表示输入框中的文本内容**

**parseFloat() 是JS中的内置函数 用于将字符串转换为浮点数(带有小数点的数字)**

**关于vscode中的声音提示(音频提示)带来的困扰**

**有一天当我在敲代码时，电脑总是错误提示，我代码一旦出错，电脑就会响一下，进行声音提示，有些时候非常烦!**

**关闭 声音提示 功能 **

**默认是 auto 在line Has Error 的选项中 将auto改为off**

**这样就不会再有声音提示了**

**串联**

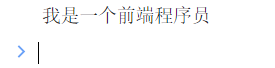
**使用 + 可以将文本的字符串 链接在一起**

const a = "我是";

        const b = "一个前端程序员";

        // const c = a + b;

        console.log(a+b);

**结果**

**也可以**

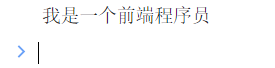
const a = "我是";

        const b = "一个前端程序员";

        const c = a + b;

        console.log(c);

**结果**

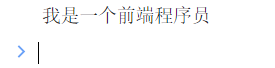
****

**复合赋值运算符**

let a = "我是";

        a += "一个前端程序员";

        console.log(a);

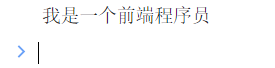
****

**等价于**

let a = "我是";

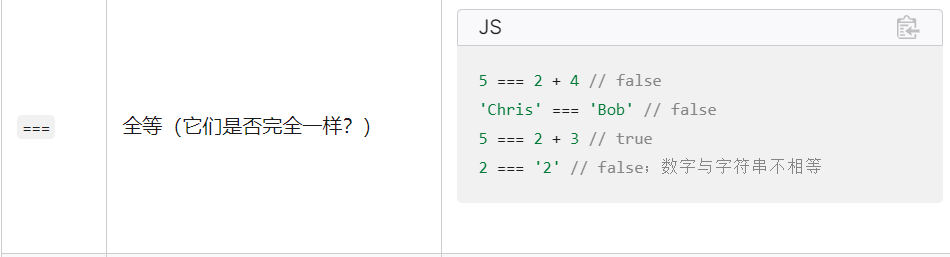
        a = a + "一个前端程序员";

        console.log(a);

****

**比较运算符**

**是否相等**

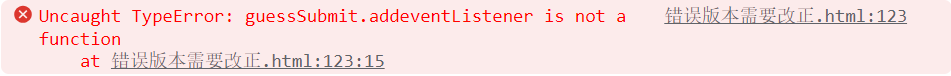
****

**是否不相等**

**是否大于**

**是否小于**

**javascript中的错误查找**

****

**解释这个代码的意思:**

**Uncaught 表示这是一个错误**

**TypeError: guessSubmit.addeventListener is not a function**

**:类型错误，在代码中的 guessSubmit.addeventListener 并不是一个函数**

**JS中的生成随机数**

**使用Math.random()这个方法可以生成0到1之间的小数**

**例如 0.345577**

**第二个方法Math.floor()舍去小数部分,所以我们利用这两个方法就可以生成随机数,问题可是会舍去小数，什么都不剩了,别急!**

**Let a = Math.floor(Math.random() \* 100);**

**Console.log(a);**

**首先哦，Math.random()随机小数，假设是一个0.456**

**然后 \* 100 这样0.456 就变成了45.6**

**然后 Math.floor()舍去小数点，变成了45**

**最后日志输出为 45 这样我们就有了随机数生成器**

**但是只有0到99的**

**Let a = Math.floor(Math.random() \* 100 + 1);**

**这样就是0到100**

**JS数组**

// 定义一个数组

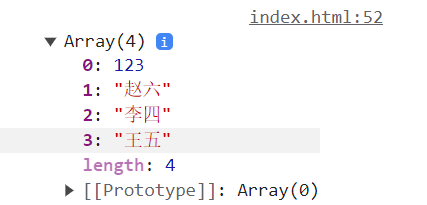
        let arr2 = [123,"张三","李四","王五"];

        // 对数组进行删除和增加

        arr2.splice(1, 1, "赵六");//删除张三 在123后面增加"赵六"

        console.log(arr2);

**结果:**

****

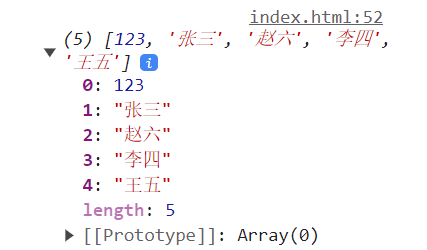
        // 定义一个数组

        let arr2 = [123,"张三","李四","王五"];

        // 对数组进行删除和增加

        arr2.splice(2, 0, "赵六");//不删除张三 在"张三"后面增加"赵六"

        console.log(arr2);

**结果:**

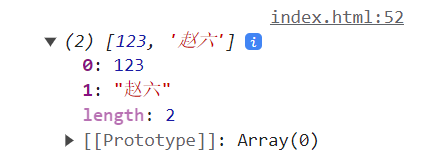
// 定义一个数组

        let arr2 = [123,"张三","李四","王五"];

        // 对数组进行删除和增加

        arr2.splice(1, 3, "赵六");//删除张三,李四,王五 增加"赵六"

        console.log(arr2);

**结果:**

**使用.length查找数组的长度**

// 定义一个数组

        let arr2 = [123,"张三","李四","王五"];

        // 对数组进行删除和增加

        arr2.splice(1, 3, "赵六");//删除张三,李四,王五 增加"赵六"

        // 使用.length 可以知道数组的长度

        console.log("arr2的长度是" + arr2.length);

**sp20231106_161622_482**

**JS中的打印方法，会激活打印机**

        print();

**结果:**

****

**JS中的输出语句**

   prompt("独裁者!必死!");

**结果:**

****

**JS中的数组使用方法**

**Let a = [1,2,3,4,5];//这是定义了一个数组**

**Let b =[“a”,”b”,”c”,”d”];//这也是定义了一个数组**

**JS中输出数组的方法**

**console.log(a[0]); //结果: 1**

**console.log(b[3]); //结果: d**

**alert(b[2]); //结果: c**

**JS中修改数组的方法**

**a[1] = 8; //将原本a中的2 修改成8**

**JS中了解数组中的长度**

**a.length; //使用length就可以知道a数组的长度**

**Console.log(a.length); //输出a数组的长度 4**

**JS中如何遍历数组**

        for(let b = 0; b<a.length; b++){

            console.log(a[b]);

        }

**JS中的多维数组**

**let c = [**

**[1,2,3],**

**[4,5,6],**

**[7,8,9]**

**];**

**输出多维数组:**

**Console.log(c[0][1]); //输出2 第一个0表示一个数组 后面的1表示第一个数组中的2**

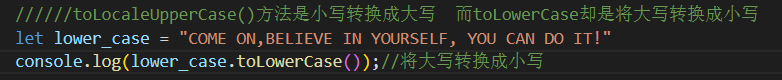
**let a = input.value;//这是字符串的形式**

**Console.log(a+2);//如果input的值是2 日志的结果是22**

**let b = parseFloat(input.value);//这是将字符串转换成数字型**

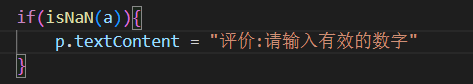
**Console.log(a+2);//如果input的值是2 日志的结果是4**

关于JS中的转换英文大小写



关于JS中的是否为非数字?

使用isNaN这是JS中的一个全局函数，用于检查一个值是否不是数字 如果不是数字返回true 是数字返回false



使用JS创建一个新的段落

let para = document.createElement(“p”);//创建一个p标签sp20231205_090011_861