1. vue是什么？

易洁，功能丰富的 Web 前端框架。

1. 创建一个 Vue 应用

创建的项目将使用基于 Vite 的构建设置，并允许我们使用 Vue 的单文件组件 (SFC)。

npm create vue@latest



输入命令之后根据命令按需求进行操作，选择是否用键盘的左右方向键。

在项目被创建后，通过以下步骤安装依赖并启动开发服务器：

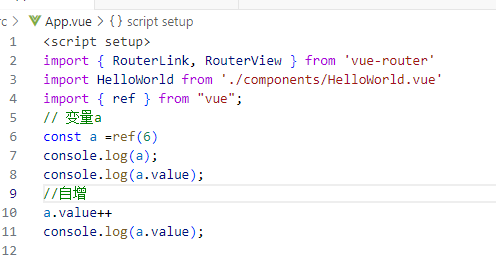
cd <your-project-name>

npm install

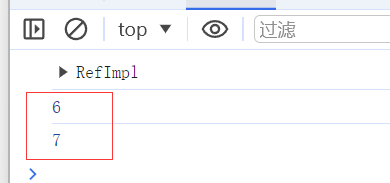
npm run dev

1. 指令是带有 v- 前缀的特殊 attribute。Vue 提供了许多内置指令，包括上面我们所介绍的 v-bind 和 v-html。一个指令的任务是在其表达式的值变化时响应式地更新 DOM。
2. 声明响应式状态

在组合式 API 中，推荐使用 ref() 函数来声明响应式状态：ref() 接收参数，并将其包裹在一个带有 .value 属性的 ref 对象中返回：



后台打印出来的结果：



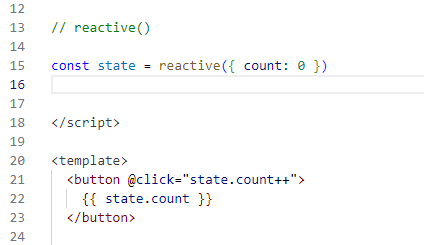
要在组件模板中访问 ref，请从组件的 setup() 函数中声明并返回它们。

Ref作用：当你在模板中使用了一个 ref，然后改变了这个 ref 的值时，Vue 会自动检测到这个变化，并且相应地更新 DOM。这是通过一个基于依赖追踪的响应式系统实现的。当一个组件首次渲染时，Vue 会追踪在渲染过程中使用的每一个 ref。然后，当一个 ref 被修改时，它会触发追踪它的组件的一次重新渲染。其中.value 属性给予了 Vue 一个机会来检测 ref 何时被访问或修改。

Dom:文档对象模型（DOM）是一个网络文档的编程接口。它代表页面，以便程序可以改变文档的结构、风格和内容。DOM 将文档表示为节点和对象；这样，编程语言就可以与页面交互。

网页是一个既可以在浏览器窗口中显示，也可以作为 HTML 源代码的文档。在这两种情况下，它都是同一个文档，但文档对象模型（DOM）的表示方式使它可以被操作。作为一个面向对象的网页表示，它可以用脚本语言（如 JavaScript）进行修改

还有另一种声明响应式状态的方式，即使用 reactive() API。与将内部值包装在特殊对象中的 ref 不同，reactive() 将使对象本身具有响应性：



reactive() 将深层地转换对象：当访问嵌套对象时，它们也会被 reactive() 包装。当 ref 的值是一个对象时，ref() 也会在内部调用它.值得注意的是，reactive() 返回的是一个原始对象的 Proxy，它和原始对象是不相等的

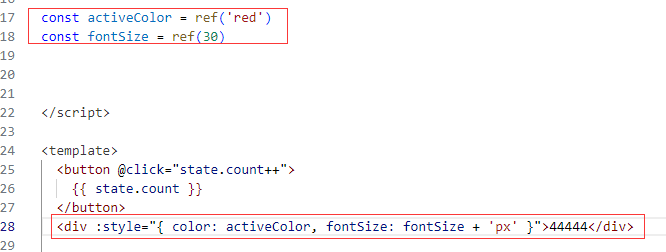


1. Class 与 Style 绑定

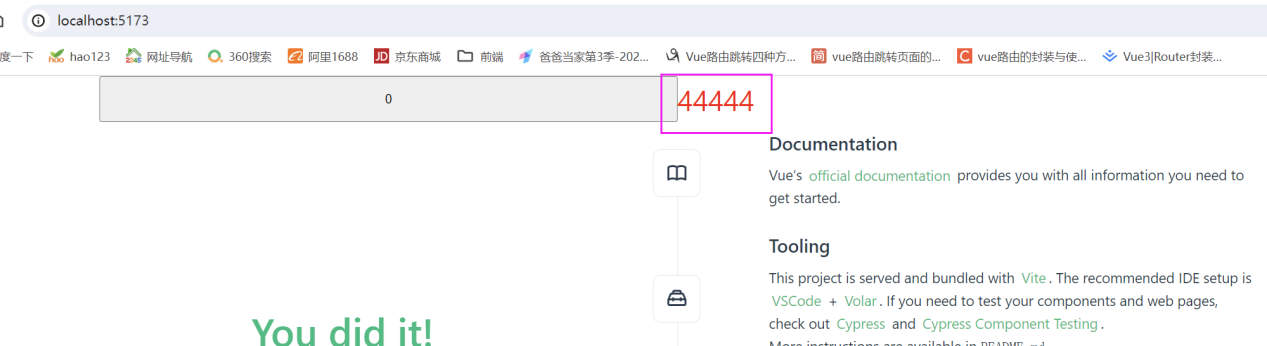
绑定内联样式

绑定对象​

:style 支持绑定 JavaScript 对象值，对应的是 HTML 元素的 style 属性：



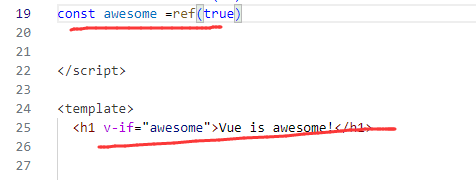
页面显示：



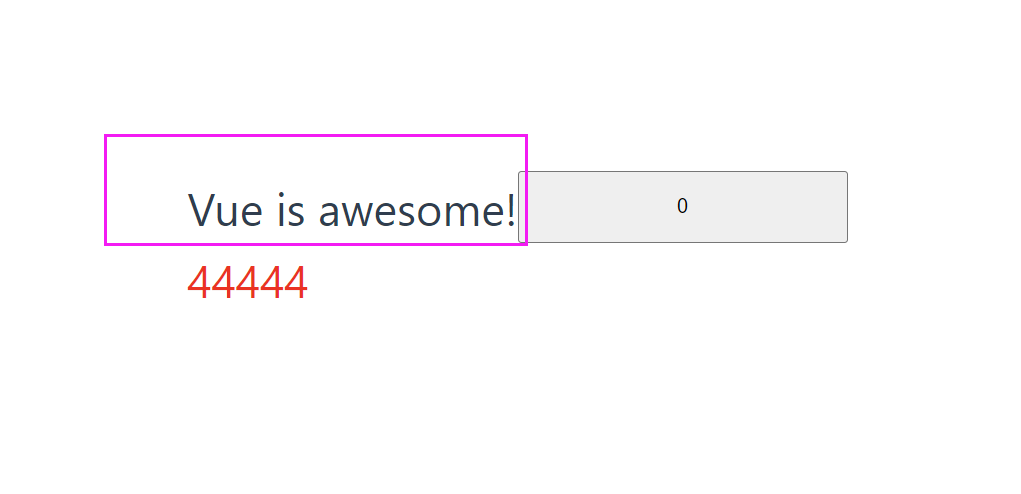
44444这个文字就得到了颜色是red 和大小是30px的属性

条件渲染

v-if 指令用于条件性地渲染一块内容。这块内容只会在指令的表达式返回真值时才被渲染，



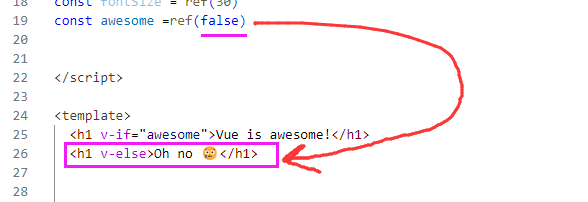
awesome设置为真，就显示这一div的内容页面显示是：



如果是false ,就不显示：

v-else​

你也可以使用 v-else 为 v-if 添加一个“else 区块”，awesome =ref(false)，就执行else的语句：



页面显示：



<template> 上的 v-if

列表渲染

v-for 指令基于一个数组来渲染一个列表。v-for 指令的值需要使用 item in items 形式的特殊语法，其中 items 是源数据的数组，而 item 是迭代项的别名：

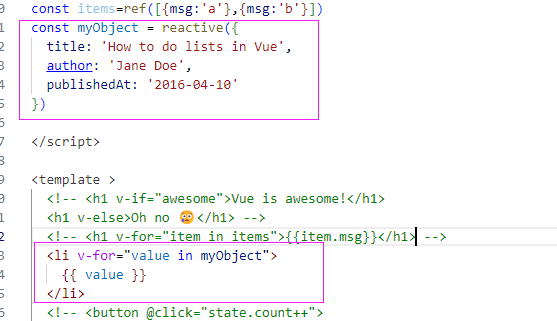


输出结果 a 和b:

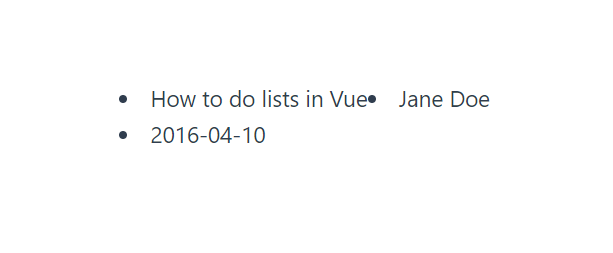


也可以使用 v-for 来遍历一个对象的所有属性。遍历的顺序会基于对该对象调用 Object.keys() 的返回值来决定。

代码如下：



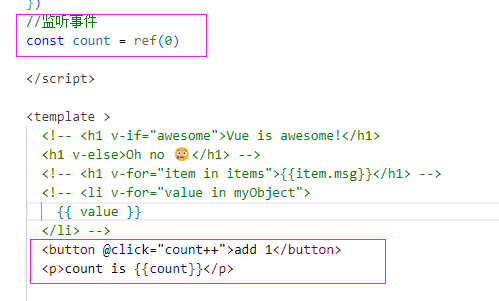
输出结果：



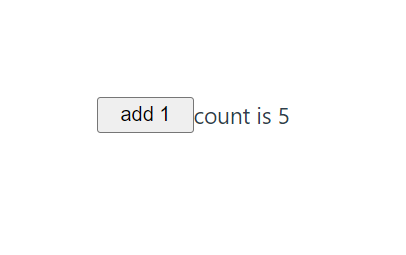
## **监听事件**

v-on 指令 (简写为 @) 来监听 DOM 事件，并在事件触发时执行对应的 JavaScript。用法：v-on:click="handler" 或 @click="handler"。

代码如下：

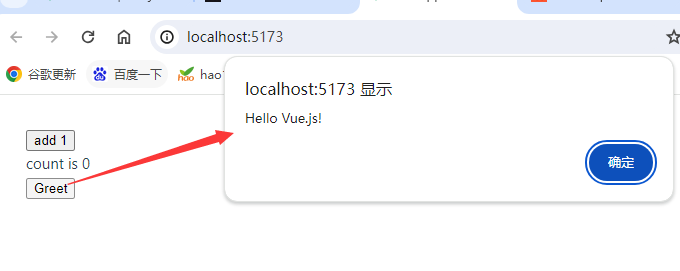


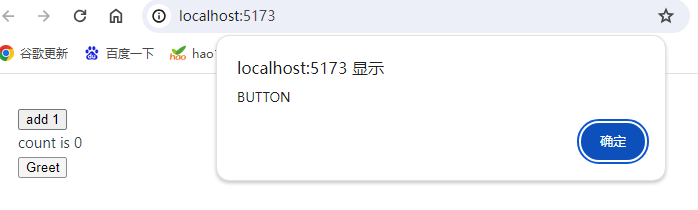
结果显示：



v-on 也可以接受一个方法名或对某个方法的调用：



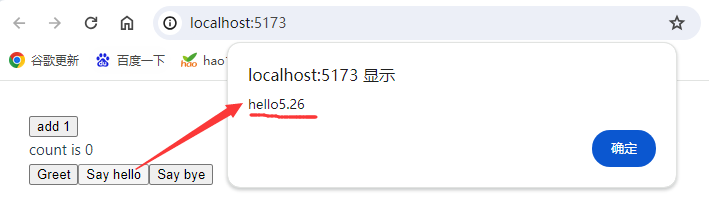


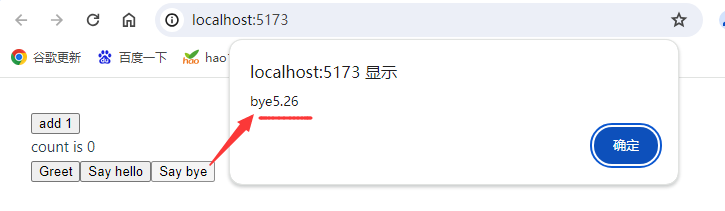


除了直接绑定方法名，你还可以在内联事件处理器中调用方法。这允许我们向方法传入自定义参数以代替原生事件：



结果显示：

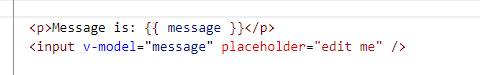




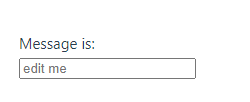
# **表单输入绑定**

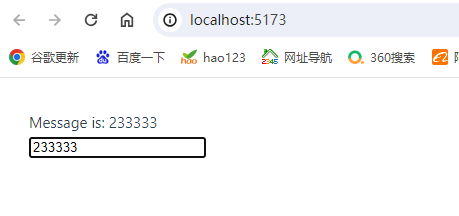
将表单输入框的内容同步给JavaScript 中相应的变量





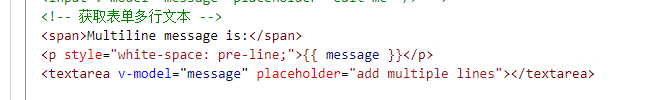
显示结果：

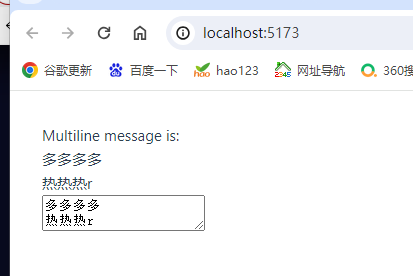




多行显示：

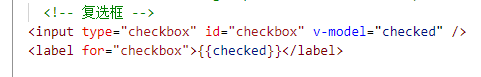
注：你可以用 v-model 指令在表单 <input>、<textarea> 及 <select> 元素上创建双向数据绑定。它会根据控件类型自动选取正确的方法来更新元素。尽管有些神奇，但 v-model 本质上不过是语法糖。它负责监听用户的输入事件以更新数据，并对一些极端场景进行一些特殊处理。





复选框





页面显示：

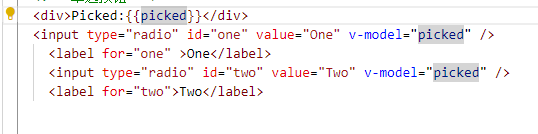


点击之后：

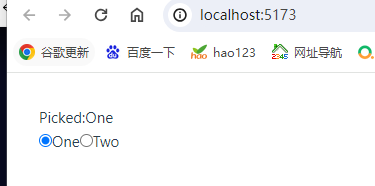


### **单选按钮**



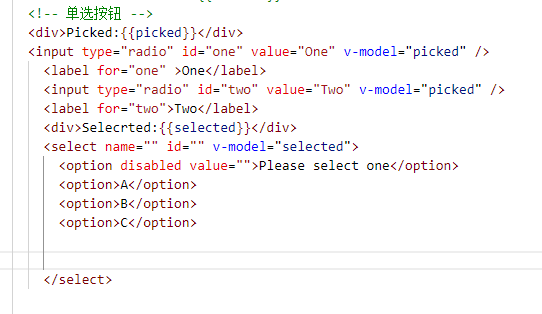


结果显示：

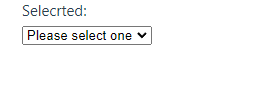


### **选择器**

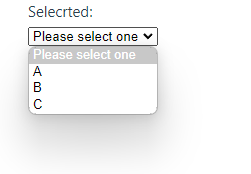




页面显示：



选择之后：



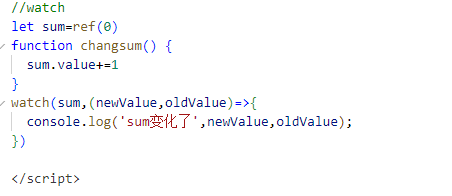
希望将该值绑定到当前组件实例上的动态数据。这可以通过使用 v-bind 来实现

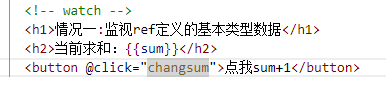
# **生命周期钩子 补**

Watch：

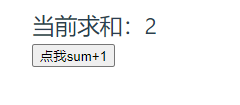


情况一:监视ref定义的基本类型数据

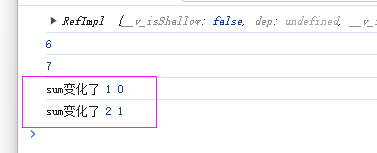




结果显示：

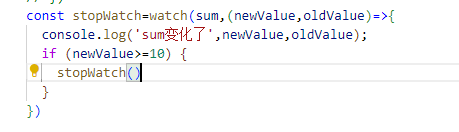


控制台输出：



如果sum的值大于10要停止监视：

代码对watch代码就行修改：



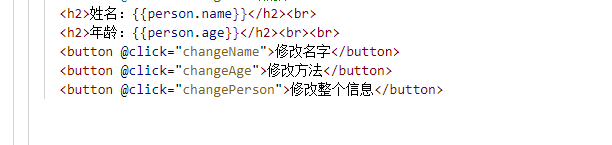
结果



现在sum=12,控制台值监视到10而已。大于10就不监视变化了

情况二:监视【ref】定义的【对象类型】数据





页面显示：



情况三

reactive定义的对象类。且默认开启了深度监视