可视化编程基础 II

2025/11/05

目录

搭建HTML框架

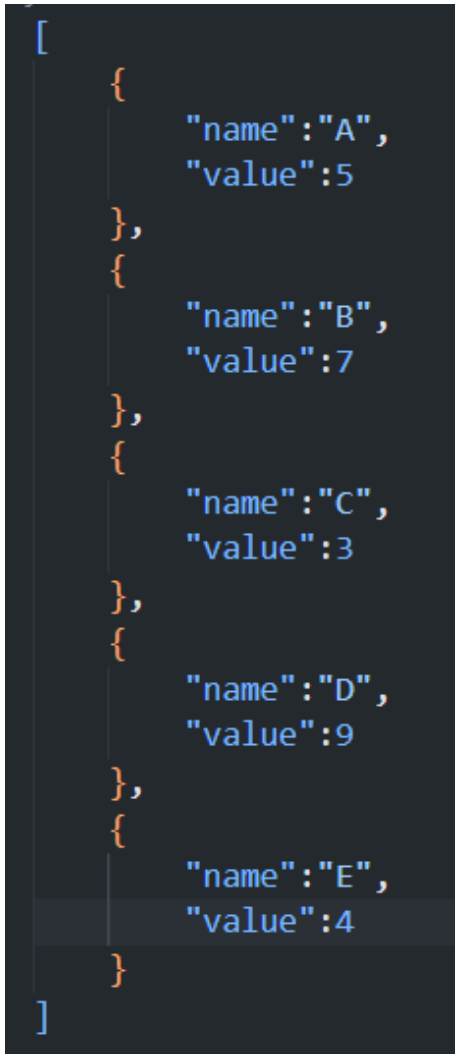
1. 新建工作目录
2. 新建index.html，告诉浏览器从这里开始解析网页
3. 搭建网页框架

```
<!DOCTYPE html>
<head>
    <meta charset="utf-8">
    <title> VisDemo </title>
</head>

<body>
    <div id='container'>
        <svg></svg>
    </div>
</body>
```

准备数据

- 前端常用的数据格式包括json和csv
- 在工作目录中新建data.json
- 随意设置一组测试数据



下载D3.js

- 使用官方提供的编译版本https://d3js.org/d3.v7.js
- 或者也可以直接从上节课下载的ScatterPlot目录中复制d3.v6.min.js
- 将下载好的d3.js放置在工作目录中，并在index.html中根据相对路径进行引用

```
<body>
    <div id='container'>
        <svg></svg>
    </div>
    <script src="https://d3js.org/d3.v7.js"></script>
</body>
```

编写脚本

- 新建main.js
- 读取窗口大小
- 设置svg大小为窗口大小
- 定义异步函数async function main()
  - 原因：d3.json是一个异步函数，只有异步函数能够以阻塞的方式调用异步函数

```
let height = window.innerHeight;
let width = window.innerWidth;

const svg = d3.select('#container')
    .select("svg")
    .attr("width", width)
    .attr("height", height);
```

编写脚本

- 使用d3.json()读取数据
- 使用d3.scaleBand()和d3.scaleLinear()分别建立x轴和y轴的比例尺
- 注意，屏幕空间的原点是左上角，y轴是向下的

```
const data = await d3.json("data.json");
// 建立比例尺
const xScale = d3.scaleBand()
    .domain(data.map(d => d.name))
    .range([0.2*width, 0.8*width])
    .padding(0.5);
const yScale = d3.scaleLinear()
    .domain([0, d3.max(data, d => d.value)])
    .range([0.8*height, 0.2*height]);
```

编写脚本

- 绘制柱状图

```
const bars = svg.selectAll("rect")
    .data(data)
    .enter()
    .append("rect")
    .attr("x", d => xScale(d.name))
    .attr("y", d => yScale(d.value))
    .attr("width", xScale.bandwidth())
    .attr("height", d => 0.8*height - yScale(d.value))
    .attr("fill", "steelblue");
```

编写脚本

- 使用d3.axis()绘制坐标轴

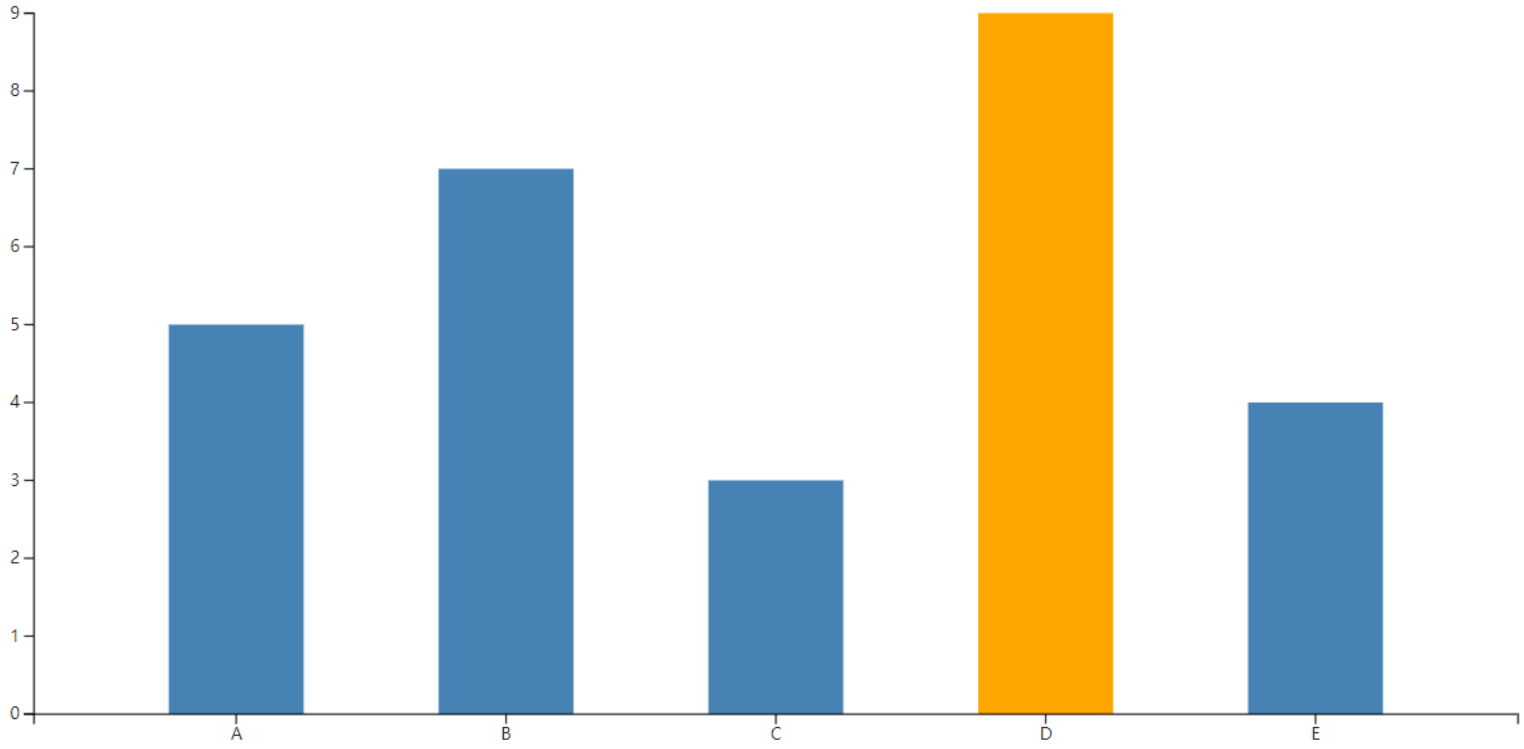
```
const xAxis = d3.axisBottom(xScale);
const yAxis = d3.axisLeft(yScale);
svg.append("g")
    .attr("transform", `translate(0, ${0.8*height})`)
    .call(xAxis);
svg.append("g")
    .attr("transform", `translate(${0.2*width}, 0)`)
    .call(yAxis);
```

编写脚本

- 使用.on绑定简单交互

```
38  bars.on("mouseover", (event, d) => {
39      d3.select(event.currentTarget)
40          .attr("fill", "orange");
41  })
42  .on("mouseout", (event, d) => {
43      d3.select(event.currentTarget)
44          .attr("fill", "steelblue");
45  });
```

效果示例



Node.js基础

- Node.js不是js文件，而是由C++编写的JavaScript运行时（类似于Python解释器）
- 相当于将浏览器中负责解释js代码的引擎单独剥离出来，使得js代码能够脱离浏览器运行
- 同时提供类似于pip和conda的包管理器npm，帮助管理js环境

Node.js安装

- Node.js官网下载：https://nodejs.org/zh-cn/download
- 选择任一LTS（Long Time Support，长期支持）版本进行下载安装
- 记得**勾选 “Add to PATH”**
- 安装完成后，运行

  node –v

  npm –v

  验证是否安装成功

Vue.js基础

- Vue.js是由尤雨溪于2014年2月发布的渐进式JavaScript框架，是当前生产环境使用最广泛的JavaScript框架之一（2020年时与谷歌Angular、脸书React形成三足鼎立格局）
- 为什么需要Vue?
  - 使用D3或原生Js时，当数据发生变化时，需要手动更新.attr() 或document.getElementById()
  - 能否让网页**自动响应**数据的变化?

Vue项目构建

- 命令行输入

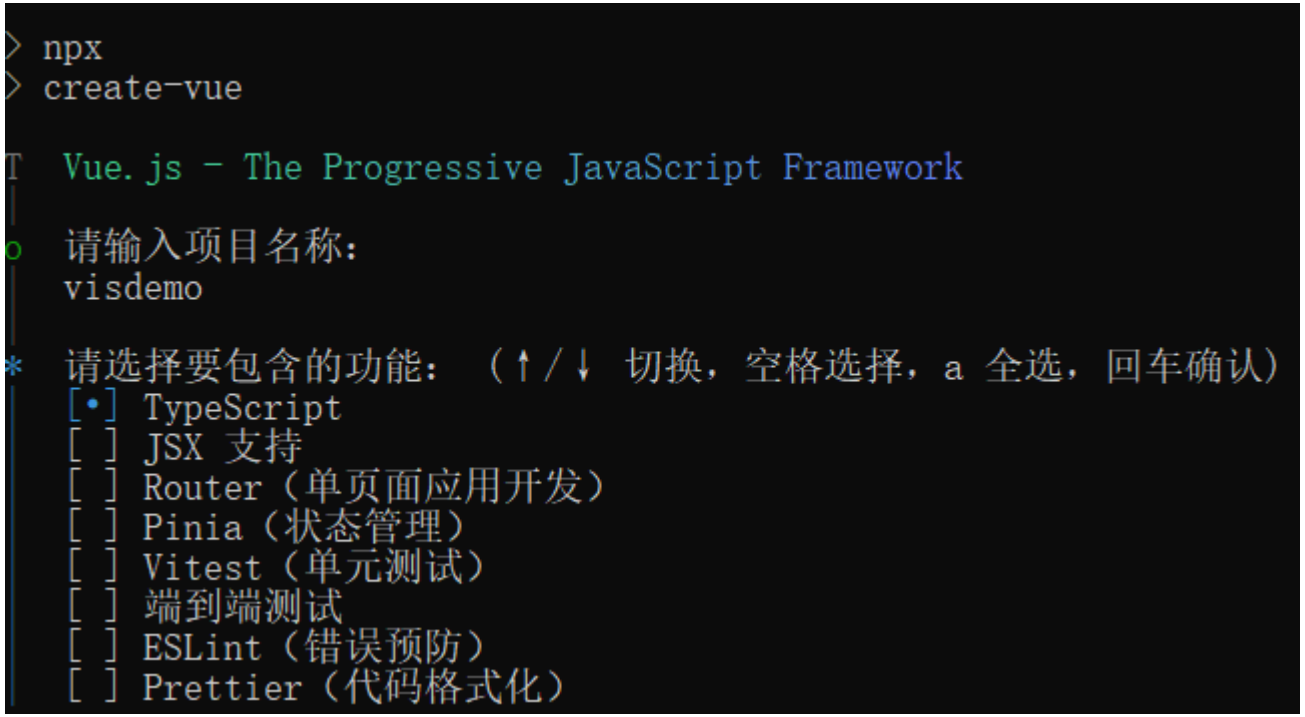
  npm create vue@latest

- 根据需要选择附加项

  常用：Pinia（跨文件变量管理）

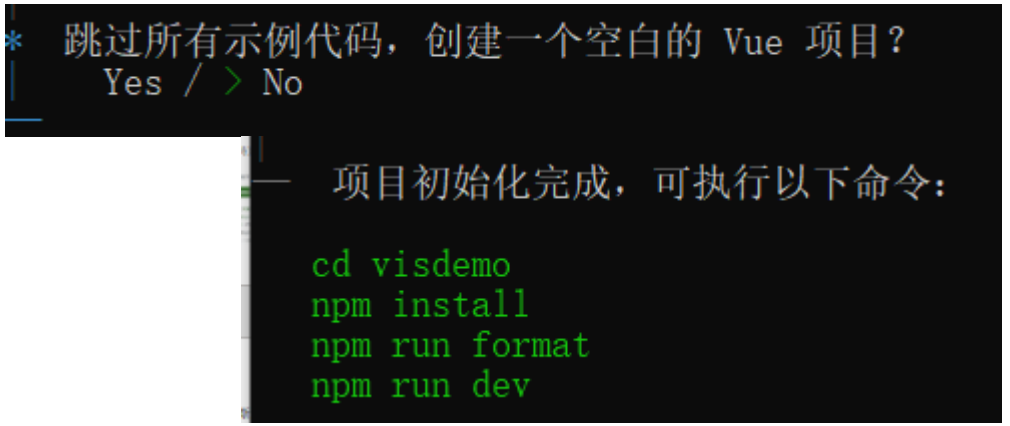
  Typescript（更严谨的显式类型声明）

  ESLint/Prettier



Vue项目构建

- 注意这里不要选Yes
- npm install：自动安装需要的包
- npm run format：将代码格式化
- npm run dev：以dev模式运行项目
- 打开命令行中输出的网址（通常是http://localhost:5173/ ）



Vue项目构建

You did it!

You've successfully created a project with Vite + Vue 3.

Documentation

Vue's official documentation provides you with all information you need to get started.

Tooling

This project is served and bundled with Vite. The recommended IDE setup is VSCode + Vue - Official. If you need to test your components and web pages, check out Vitest and Cypress / Playwright.
More instructions are available in `README.md`.

Ecosystem

Get official tools and libraries for your project: Pinia, Vue Router, Vue Test Utils, and Vue Dev Tools. If you need more resources, we suggest paying Awesome Vue a visit.

Community

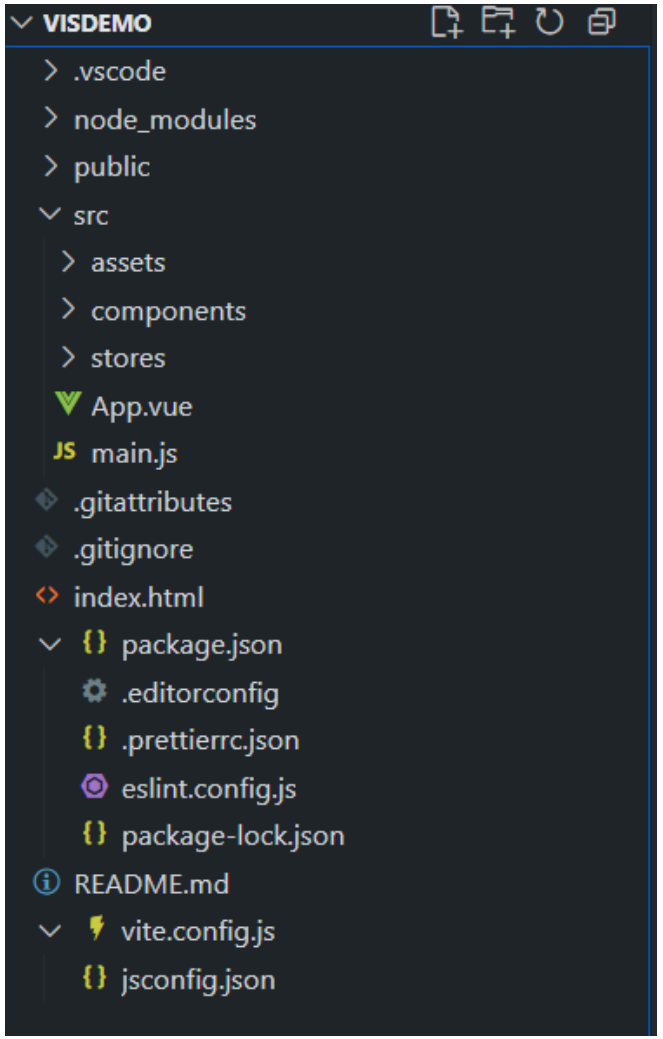
Got stuck? Ask your question on Vue Land (our official Discord server), or StackOverflow. You should also follow the official @vuejs.org Bluesky account or the @vuejs X account for latest news in the Vue world.

Support Vue

As an independent project, Vue relies on community backing for its sustainability. You can help us by becoming a sponsor.

目录结构

- node_modules : 项目引用的包
- public : 项目的外部资源，包括数据、图片等
- src : 项目的源代码
- main.js : 项目的主要程序入口
- index.html : 网页的主要入口



文件结构

- <script setup> : js代码，负责逻辑
- <template> : html代码，负责结构
- <style scoped> : css代码，负责样式

```vue
<script setup>
defineProps({
  msg: {
    type: String,
    required: true,
  },
})
</script>

<template>
  <div class="greetings">
    <h1 class="green">{{ msg }}</h1>
    <h3>
      You've successfully created a project with
      <a href="https://vite.dev/" target="_blank" rel="noopener">Vite</a> +
      <a href="https://vuejs.org/" target="_blank" rel="noopener">Vue 3</a>.
    </h3>
  </div>
</template>

<style scoped>
h1 {
  font-weight: 500;
  font-size: 2.6rem;
  position: relative;
  top: -10px;
}

h3 {
  font-size: 1.2rem;
}

.greetings h1,
.greetings h3 {
  text-align: center;
}
</style>
```

响应式变量定义

- const count = ref(0)
- ref关键字为变量添加了一个跟踪器，当count的值(count.value)发生变化时，vue会自动追踪所有引用这一变量的html元素，并动态更新
- 在template中这样引用：{{count}}

响应式变量定义

- const doubleCount = computed(()=>count.value * 2)
- computed关键字使一个变量随着参与他运算的所有响应式变量的变化而自动更新

手动监听变量

- watch关键字允许开发者手动监听响应式变量
- watch(()=>count.value,(newVal,oldVal)=>{

  console.log(oldVal,newVal)

  })

Vue的生命周期

- 一个Vue文件就是一个组件，其在被挂载到html网页上时被注册，在其被关闭时被销毁
- Vue的全生命周期包括:
  - 🧱 创建前/后　onBeforeMount()、onMounted()
  - 🔄 更新前/后　onBeforeUpdate()、onUpdated()
  - ❌ 卸载前/后　onBeforeUnmount()、onUnmounted()
- 通过这些钩子，开发者可以指定在特定阶段时进行的操作

Vue的template语法

- v-for : 允许根据列表生成网页结构
  - <g v-for="item in data">
- v-if/v-show : 允许根据变量值控制元素的显隐
  - <img v-if = "displayImage">
- :attr : 允许根据响应式变量自动改变元素的属性值
  - <svg :height = "height" :width = "width">
- @click/@dbclick/@mouseover/@mouseout:快捷绑定事件函数
  - <button @click = "()=>count++">

更多练习

- https://play.vuejs.org/ 在线vue练习平台
- https://cn.vuejs.org/ 官方中文文档
- https://vueschool.io/ 项目驱动课程练习
- 其他视频资源