加餐丨进阶篇思考题答疑

2022-12-26 杨文坚 来自北京

《Vue 3 企业级项目实战课》

课程介绍 >

讲述：杨文坚

时长 00:40　大小 626.33K

你好，我是杨文坚。

在进阶篇中，我们着手打造自己的 Vue.js 3.x 自研组件库，分别开发了主题方案、基础组件比如按钮组件、动态渲染组件、布局组件、表单组件，为我们开发业务组件库和打造一个运营搭建平台打好基础。

最后，出于组件库代码的“稳定性”和“健壮性”考虑，我们基于 Vitest，实现了 Vue.js 3.x 组件库的单元测试。

今天我们会对进阶篇做一次统一的答疑，如果还有疑问，也欢迎你留言。

08

课程：08 从零搭建自研的 Vue 组件库

提问：组件库的按需加载实现方式，还有其它的方案吗？

有其它方案的。思考这个问题，要理清楚一点，这里的“按需加载”指的是，在编译过程中，根据实际项目“import”的组件，进行独立 JavaScript 文件加载，没有“import”到的组件就不加载。

那么在编译过程中，我们可以在 JavaScript 代码的语法树（AST）上做处理，Ant Design 团队就开发了一个 babel 插件，叫“babel-plugin-import”，可以根据实际组件库的构建结果文件的目录结构，进行自定义的配置组件的按需加载，具体插件 npm 链接在这里：https://www.npmjs.com/package/babel-plugin-import 。

09

课程：09 设计组件库的主题方案和基础组件

提问：为什么主题控制只考虑颜色，不考虑组件的尺寸的形状控制呢？

因为主题的影响是全局性的，所有用到该主题尺寸配置的组件，都会受到影响。如果组件的尺寸（例如高度、宽度等）可以被主题全局化自定义控制，那么就会带来布局的变化。设想一下，现在，有个标题组件固定尺寸大小显示，被全局的配置自定义修改增加了高度，那么整个标题组件所在的父组件可能会被“撑开”，高度可能会“溢出”，带来不可预测的问题。

这里“尺寸不允许自定义”只是相对的限制，如果要针对某个组件的尺寸做自定义配置，我们可以设计这个组件的 Props 或者其它 API，来提供单独组件的尺寸自定义修改。

10

课程：10 实现 Vue 的动态渲染组件

提问：Vue.js 3.x 动态渲染组件是通过创建一个新的 Vue.js 应用来渲染组件的，跟页面原有的 Vue.js 应用是互相独立的，那么如何实现这两个 Vue.js 应用的数据通信呢？

可以通过“事件订阅和发布”的方式来进行组件之间的互相通信。具体实现方式也很简单：

- 第一步，实现一个事件订阅和发布的类 EventEmitter。这个具体代码实现网上有很多，你可以自行搜索。全局共用一个 EventEmitter 创建出来的对象 eventHub；

- 第二步，在组件 A 里用 eventHub 注册一个事件“callA”；
- 第三步，在组件 B 里的某个操作里，用 eventHub 触发“callA”的事件，来进行对 A 组件的通信。

伪代码如下所示：

```
import EventEmitter from "./xxx/xxx/EventEmitter"
const eventHub = new EventEmitter();

const A = function() {
  // 注册全局的事件
  eventHub.on("callA", (data) => {
    console.log("A组件被触发事件了，接收到数据", data)
  })

  // ...
}

const B = function() {
  // ...
  return {
    connectA() {
      // 触发事件 "callA" 去跟A组件通信
      // 并传入数据  { abc: 123 }
      eventHub.trigger("callA", { abc: 123 })
    }
  }
}

const b = B();
// 跟A组件通信
b.connectA();
```

11

课程：11 实现自研组件库的布局方案

提问：如何实现一套布局组件方式，同时兼容 PC 页面和移动端页面的布局？

想要用一套布局代码，实现 PC 端和移动端兼容，最通用的解决方案就是“Web 网页的自适应布局”和“Web 网页的响应式布局”。

“Web 网页的自适应布局”就是网页内容的尺寸是随着屏幕的大小变化，通常是用“比例”作为布局尺寸，根据不同尺寸的屏幕，做一定的比例处理。例如 CSS 尺寸单位用 rem、em、vh、vw 等相对单位。同时，也会结合 CSS 媒体查询 @media，进行监听屏幕尺寸变化，做对应 CSS 样式变化。

“Web 网页的响应式布局”核心是基于 CSS 媒体查询 @media ，设计几种尺寸阶梯的样式布局，用一套代码，囊括多种屏幕尺寸的布局 CSS。当 @media 媒体查询检测到屏幕尺寸落在哪个区间里，就用对应区间的 CSS 样式进行显示。

12

课程：12 开发受控的表单组件

提问：表单组件除了劫持代理“submit”事件，还有其它的方式来管理表单提交数据的操作吗？

表单组件劫持代码“submit”事件来管理表单提交，核心是利用 HTML 的原生事件特性。如果要用其它方式，那就是不用表单原生提交事件，可以自行封装处理表单提交操作。

自行封装管理表单提交数据，意味着受控表单组件全程管理数据，不再利用 HTML 原生表单时间，提交表单时候，点击按钮或者快捷键提交，都统一交于受控组件来人工管理的“公共数据”，最后发起表单请求。

13

课程：13 开发动态渲染表单组件

提问：动态表单能实现多种表单数据组件的渲染，那不同表单数据组件，能否做联动操作的功能配置？

可以做联动处理，例如表单里有 A 字段和 B 字段，其中，B 字段要等待 A 字段选择了数据，才能进行编辑。这个过程，其实类似受控组件里的“校验”操作，只不过在校验过程中传入整个表单数据来进行判断。

所以，要实现动态表单组件的“数据组件的联动功能”，核心是扩展“受控表单字段组件”的校验功能，校验逻辑要加上判断其他字段数据的能力。

14

课程：14 代码单元测试，轻松地保证自己的代码质量

提问：这节课都是模板语法写的单元测试，Vue.js 3.x 的 JSX 语法单元测试要怎么写呢？

JSX 语法的单元测试，其实比模板语法的单元测试更加灵活，因为不需要把组件测试内容放在独立的 Vue 文件里。

JSX 语法测试的组件，在同个测试文件里就能进行测试，如下代码所示：

复制代码

```
import { describe, test, expect } from 'vitest';
import { nextTick, defineComponent, ref } from 'vue';
import { mount } from '@vue/test-utils';
import { Button } from '../../src';

describe('Button', () => {
  test('click event', async () => {
    const ButtonTest = defineComponent({
      setup() {
        const num = ref(123);
        const onClick = () => {
          num.value += 1;
        };
        return {
          num,
          onClick
        };
      },
      render(ctx) {
        const { num, onClick } = ctx;
        return (
          <div>
            <div class="display-text">当前数值={num}</div>
            <Button class="btn-add" onClick={onClick}>
              点击加1
            </Button>
          </div>
        );
      }
    });

    const wrapper = mount(ButtonTest, { props: {} });
    const textDOM = wrapper.find('.display-text');
    const btnDOM = wrapper.find('.btn-add');
    expect(textDOM.text()).toBe('当前数值=123');
```

```
    btnDOM.trigger('click');
    await nextTick();
    expect(textDOM.text()).toBe('当前数值=124');
  });
});
```


下一讲我们进入实战篇，把低代码搭建页面的技术概念融入课程中，搭建起一个企业级项目，搭建过程中，我们会分解成一个个简单易懂的技术知识点，并尝试用已学的、合适的实现方式优雅解决。下一讲见。

上一篇 14丨代码单元测试：如何轻松地保证自己的代码质量？

下一篇 15丨定制运营拖拽组件：如何实现运营搭建页面的拖拽功能？

精选留言

写留言

由作者筛选后的优质留言将会公开显示，欢迎踊跃留言。