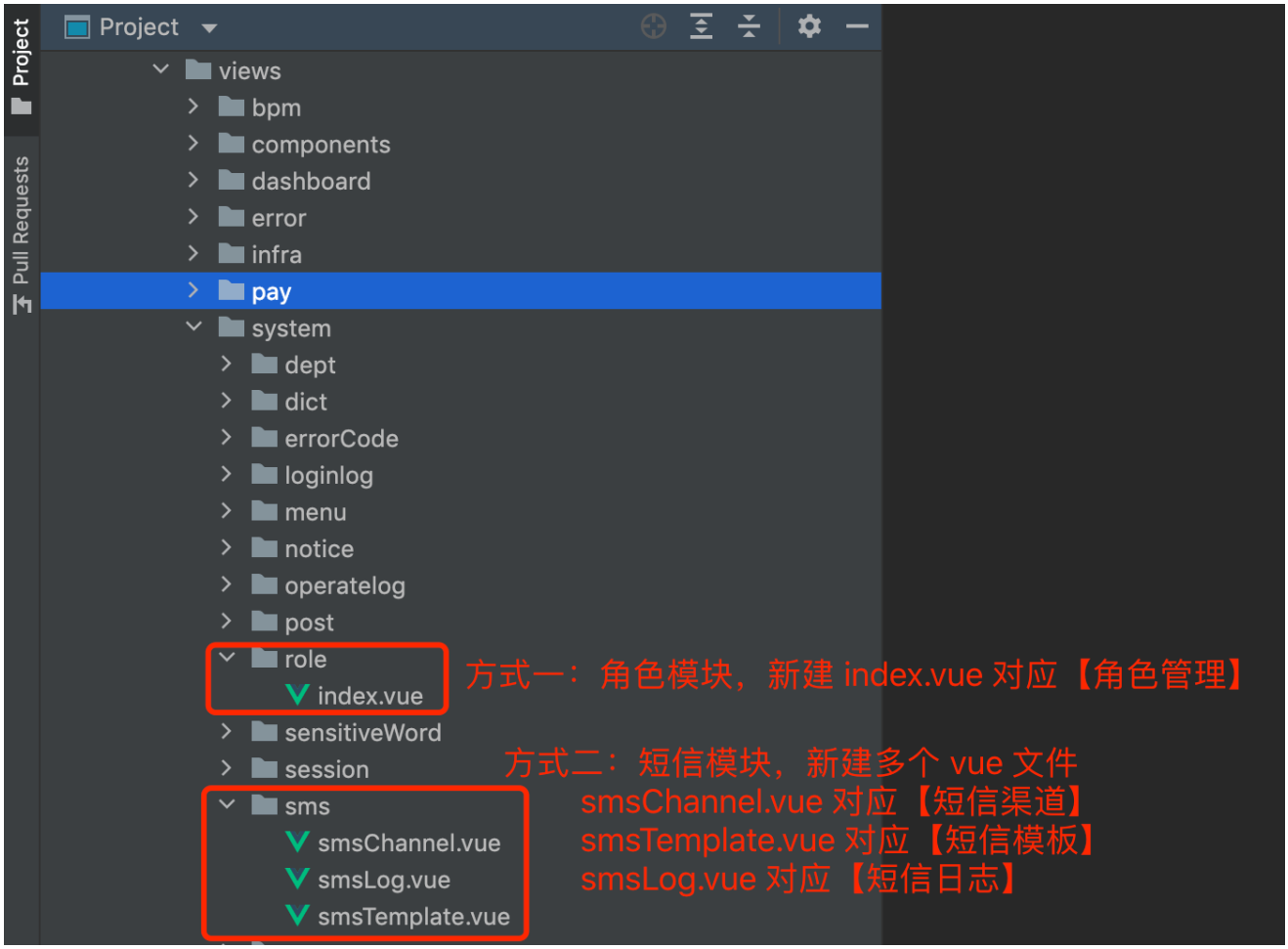
芋道源码 2022-04-17

开发规范

1. view 页面

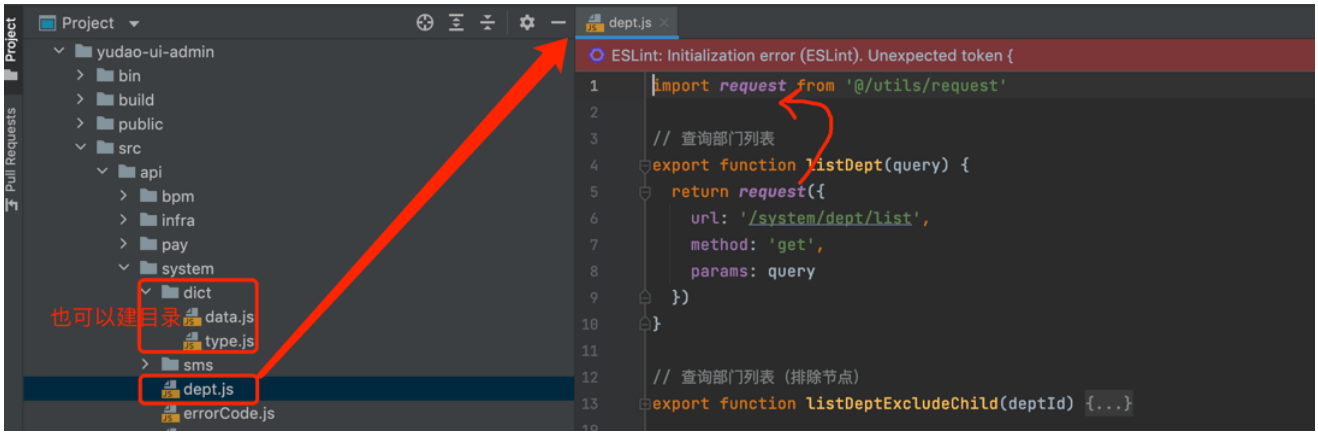
在 `@views` 目录下，每个模块对应一个目录，它的所有功能的 `.vue` 都放在该目录里。



一般来说，一个路由对应一个 `.vue` 文件。

2. api 请求

在 `@/api` 目录下，每个模块对应一个 `.api` 文件。



每个 API 方法，会调用 `request` 方法，发起对后端 RESTful API 的调用。

2.1 请求封装

`@/utils/request` 基于 axios 封装，统一处理 GET、POST 方法的请求参数、请求头，以及错误提示信息等。

2.1.1 创建 axios 实例

- `baseURL` 基础路径
- `timeout` 超时时间

▶ 实现代码

2.1.2 Request 拦截器

- `Authorization` 、 `tenant-id` 请求头
- GET 请求参数的拼接

▶ 实现代码

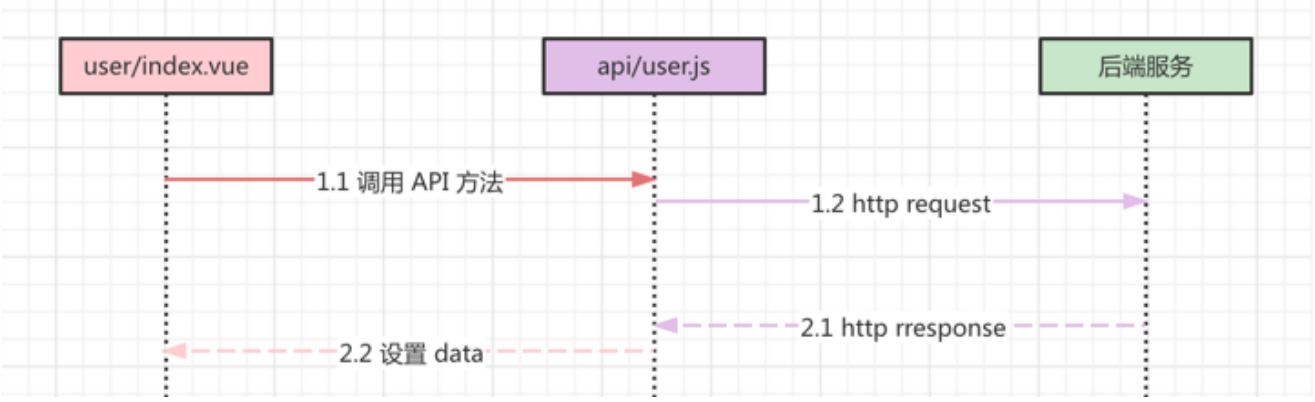
2.1.3 Response 拦截器

- Token 失效、登录过期时，跳回首页
- 请求失败，Message 错误提示

▶ 实现代码

2.2 交互流程

一个完整的前端 UI 交互到服务端处理流程，如下图所示：



以 [系统管理 -> 用户管理] 菜单为例，查看它是如何读取用户列表的。代码如下：

```
// ① api/system/user.js
import request from '@/utils/request'

// 查询用户列表
export function listUser(query) {
    return request({
        url: '/system/user/page',
        method: 'get',
        params: query
    })
}

// ② views/system/user/index.vue
import { listUser } from "@/api/system/user";

export default {
  data() {
    userList: null,
    loading: true
  },
  methods: {
    getList() {
      this.loading = true
      listUser().then(response => {
        this.userList = response.rows
        this.loading = false
      })
    }
  }
}
```

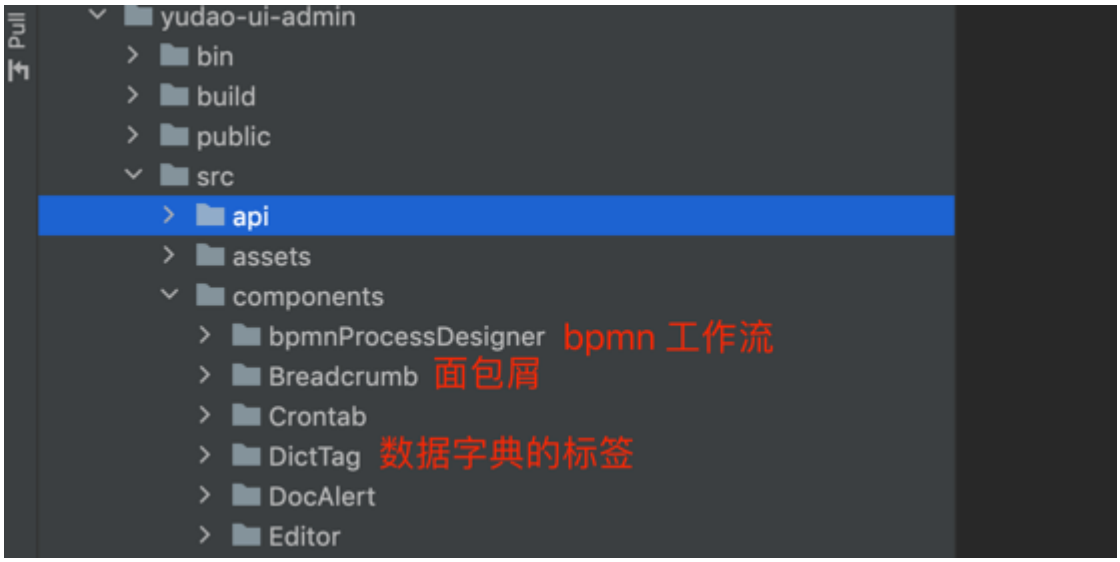
2.3 自定义 baseURL 基础路径

如果想要自定义的 baseURL 基础路径，可以通过 baseURL 进行直接覆盖。示例如下：

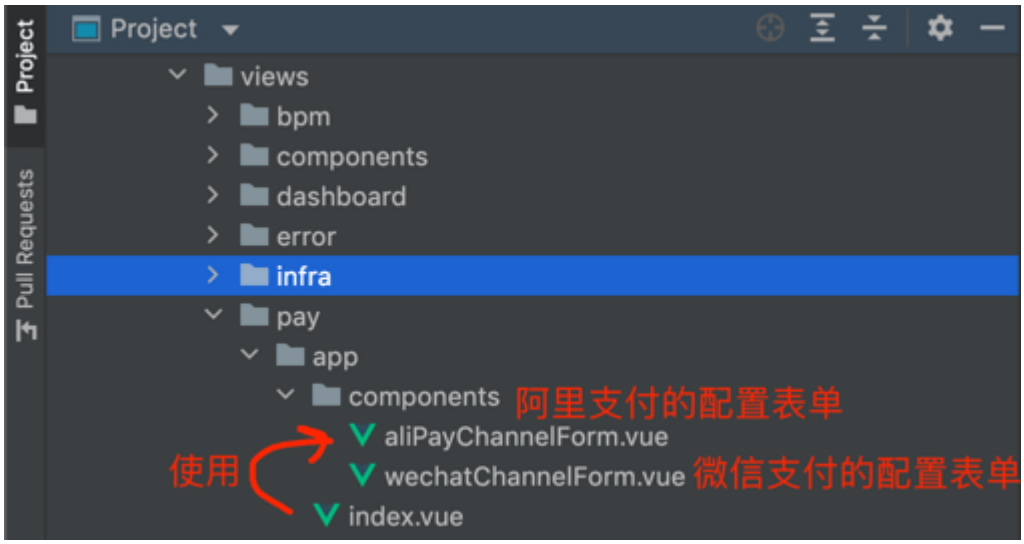
```
export function listUser(query) {
    return request({
        url: '/system/user/page',
        method: 'get',
        params: query,
        baseURL: 'https://www.iocoder.cn' // 自定义
    })
}
```

3. component 组件

① 在 @/components 目录下，实现**全局**组件，被所有模块所公用。例如说，富文本编辑器、各种各搜索组件、封装的分页组件等等。

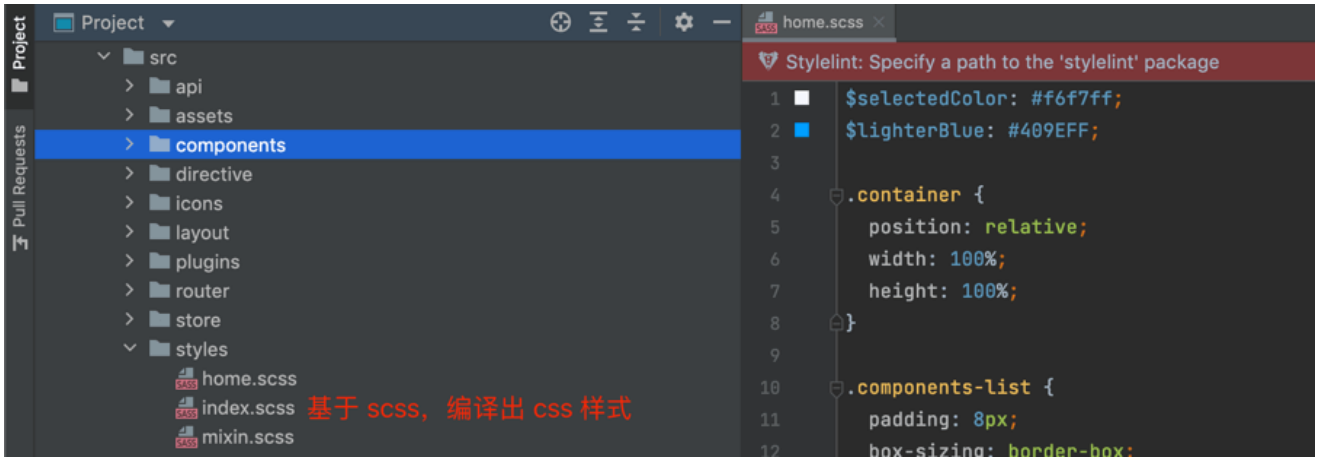


② 每个模块的业务组件，可实现在 views 目录下，自己模块的目录的 components 目录下，避免单个 .vue 文件过大，降低维护成功。例如说，@/views/pay/app/components/xxx.vue 。

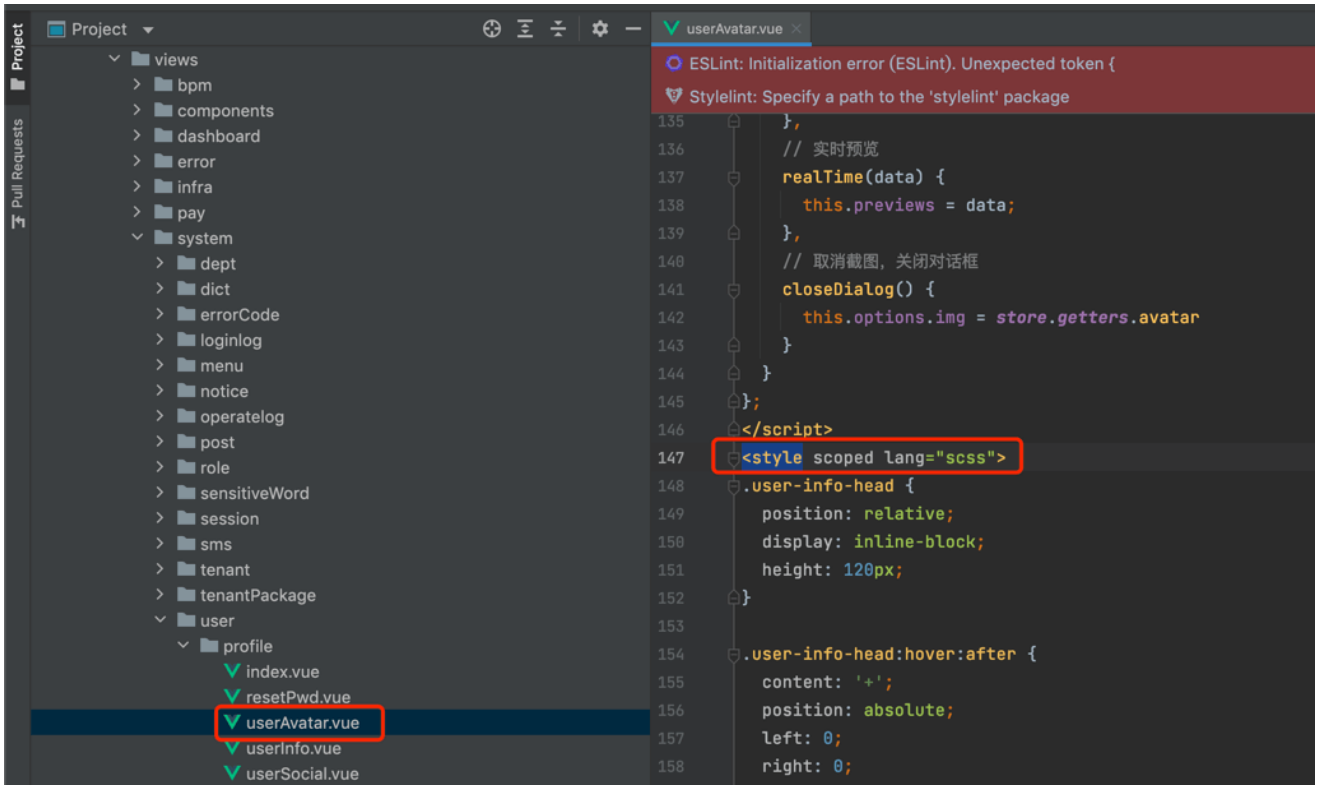


4. style 样式

① 在 `@/styles` 目录下，实现**全局**样式，被所有页面所公用。



② 每个 `.vue` 页面，可在 `<style />` 标签中添加样式，注意需要添加 `scoped` 表示只作用在当前页面里，避免造成全局的样式污染。



← 服务监控 菜单路由→