ChatGPT

Vue 3 实战规范总结手册

本文简明汇总了自前端实际项目中流行的 Vue 3 开发规范、优化方法和代码编写模式，适合固定团队或个人定制 Vue 项目统一规范。

一、基础经验

1.1 箭头函数

- 优先使用 `const + 箭头函数`
- 避免混用 `function` 和箭头函数`

1.2 变量描述

- 优先使用 `let` / `const`，避免变量提升
- `const` 用于常量、不变值的变量

1.3 异步请求

- 优先使用 `async/await + try/catch`
- 避免异步回调地狱
- 名称推荐带 `Api` 结尾，如 `getUserListApi()`

1.4 应用 Promise 并发 / 竞速

- 并发: `Promise.all`
- 竞速: `Promise.race`
- 如需有失败耗时详情，使用 `Promise.allSettled`

1.5 响应式变量

- 大数据 / 深层对象，使用 `shallowRef` / `shallowReactive`
- 避免过度响应性干扰

1.6 组件 / 方法单一职责

- 一个组件/方法只做一件事

1.7 名称规范

- 组件、文件、`<script setup name='XX'>` 都用 **PascalCase**
- 运行时 name 输出对象要唯一

二、监听器 / Hooks

2.1 `watchEffect`

- 较适合简单部分边效影响
- 遇到异步操作时不符合预期，优先考虑 `watch`

2.2 `watch` 配合 `effectScope`

- 用于惯性定义同一区块的一组 watch 或 effect
- 在 `onBeforeUnmount` 中手动 `.stop()` 放缓

三、Hooks 分离思想

3.1 较好的 hooks 结构:

```
const useEditModal = () => {
  const isShow = ref(false);
  const show = () => {}
  const close = () => {}
  const submit = () => {}
  return { isShow, show, close, submit }
}
```

3.2 合理投影 expose 方法

```
expose(new Proxy({}, {
  get(_, key) {
    return refEl.value?.[key];
  },
  has(_, key) {
    return key in refEl.value;
  }
}))
```

四、组合式 API 优先顺序

```
// <script setup>
// 1. import
// 2. defineProps / defineEmits
// 3. refs / reactives
// 4. computed
// 5. watch / watchEffect
// 6. 函数区
```

```
// 7. lifecycle
// 8. defineExpose
```

五、逻辑分支优化

5.1 多 if else 分支

- 用调度表 (map) 解耦

```
const map = {
 1: handle1,
 2: handle2,
 3: handle3,
 default: handleDefault
}
(map[type] || map.default)()
```

5.2 嵌套分支

- 推荐早返回(return early)
- 或抽出方法: `if (isValidType()) {}`

六、清理旧代码

- 去掉过期注释、console.log/debugger
- 单文件不超过600行（充分分类、分组件、分 hooks、分配置、分样式、分方法、分 utils）

七、异步组件 / 路由應用

7.1 defineAsyncComponent

```
const AdminPage = defineAsyncComponent(() => import('./AdminPage.vue'))
```

7.2 路由應用加载分离

```
const UserDetail = () => import('./views/UserDetail.vue')
{
 path: '/user/:id',
 component: UserDetail
}
```

八、新语法操作符

- `??` 空值合并
- `?.` 可选链表达式
- `??=` 空值赋值符

九、注释规范

9.1 文件头注释

```
/*
 * @FileDescription: 表明用途
 * @Author: dev
 * @Date: yyyy-mm-dd
 */
```

9.2 方法注释 (JSDoc)

```
/**
 * @description: 获取列表
 * @param {Object} user
 * @return void
 */
```

9.3 行内 / 分区注释

- `// 操作表格列表排序`
- `// #region X处理` ... `// #endregion`

9.4 新增 / bugfix

```
// FEAT-001: 实现 XX 功能 (LMX-2024-06-05)
// BUGFIX-001: 修复 XX 问题 (LMX-2024-06-05)
```

十、目录组织推荐

```
custom_module
 ├──── api
 ├──── components/modules
 ├──── composable
 ├──── methods/hooks
 ├──── functional
 ├──── config
```

```
├── styles
├── utils
├── index.vue
├── .pubrc.js
└── README.md
```

十一、性能优化

- 清理余余的代码
- 组件抽离，功能处理抽离
- 优先选用 lodash-es
- 合理分割打包 chunk
- 合理路由分离
- 合理缓存 local/session/indexedDB
- 合理分段请求、虚拟滑动
- 避免意外的响应性跳动
- 避免 key=index 惊动