#### **语法规**

#### **前端开发规范**

标红的部分注意掌握，好统一风格

1.组件名应为多个单词，见名知意，首字母大写的驼峰命名法，根组件和Vue内置组件除外。

​ 例：TodoItem

​ template 页面引用可以是 也可以是

2.组件的 data 必须是一个函数。

​ 例：

data () { return { foo: 'bar' } }

3.Prop 的定义应该尽量详细，至少需要指定其类型。类型注明更便于后续更改。

​ 例：props: { status: String }或者更为详细 props:{ status: { type: String, default: 'xxx' }}

4.v-for 循环遍历数据必须配置键值 :key 以便维护内部组件及其子树的状态。

5.避免v-if 和 v-for 用在一起，永远不要把 v-if 和 v-for 同时用在同一个元素上。当 Vue 处理指令时，v-for 比 v-if 具有更高的优先级。如需使用可以使用computed 计算属性过滤后使用 或者 用template 标签语法 又或者 挂载在其父元素DOM上 分开使用

​ 例：

<div v-for="user in users" :key="user.id">

<template v-if="user.status">

<span>

{{user.title}}

</span>

</template>

<template v-else>

<span>

{{user.desc}}

</span>

</template>

</div>

6.为组件样式设置作用域，避免全局污染 。这让覆写内部样式更容易：使用了常人可理解的 class 名称且没有太高的选择器优先级，而且不太会导致冲突。对于应用来说，顶级 App 组件和布局组件中的样式可以是全局的，但是其它所有组件都应该是有作用域的。

​ 例：

7.私有property名命名规则，使用模块作用域保持不允许外部访问的函数的私有性

​ 例：$\_yourPluginName\_xxx

8.单文件组件的文件名应该要么始终是单词大写开头 (PascalCase)，要么始终是横线连接 (kebab-case)。

​ 例：components/

​ | MyComponent.vue

​ components/

​ | my-component.vue

9.基础组件名，应用特定样式和约定的基础组件 (也就是展示类的、无逻辑的或无状态的组件) 应该全部以一个特定的前缀开头，比如 Base、App 或 V

​ 例：components/

​ |- BaseButton.vue

​ |- BaseTable.vue

​ |- BaseIcon.vue

10.单例组件名，单例组件指只应该拥有单个活跃实例的组件，其命名应该以 The 前缀命名，以表示其唯一性。

​ 例：

components/

|- TheHeading.vue

|- TheSidebar.vue

11.紧密耦合的组件名，和父组件紧密耦合的子组件应该以父组件名作为前缀名

​ 例：

components/

|- TodoList.vue

|- TodoListItem.vue

|- TodoListItemButton.vue

12.组件名中的单词顺序，应该以高级别的（功能性和实用性，如search）单词开头，一般运用于非常大型的项目方便查找使用

​ 例：

components/

|- SearchButtonClear.vue

|- SearchButtonRun.vue

|- SearchInputQuery.vue

|- SearchInputExcludeGlob.vue

|- SettingsCheckboxTerms.vue

|- SettingsCheckboxLaunchOnStartup.vue

13.自闭合组件，在单文件组件、字符串模板和 JSX 中没有内容的组件应该是自闭合的

​ 例：

<!-- 在单文件组件、字符串模板和 JSX 中 -->

<MyComponent/>

<!-- 在 DOM 模板中 -->

<my-component></my-component>

14.Prop 名大小写，在声明 prop 的时候，其命名应该始终使用 camelCase, 在模板和 JSX 中应该始终使用 kebab-case。

例：

// js

props: {

greetingText: String

}

// html

<WelcomeMessage greeting-text="hi"/>

多个 attribute 的元素 ，应该分多行撰写，每个 attribute 一行。

例：

<img

src="https://vuejs.org/images/logo.png"

alt="Vue Logo"

>

<MyComponent

foo="a"

bar="b"

baz="c"

/>

模板中简单的表达式，组件模板应该只包含简单的表达式，复杂的表达式则应该重构为计算属性或方法。

例子：

---HTML---

<!-- 在模板中 -->

{{ normalizedFullName }}

---JS---

// 复杂表达式已经移入一个计算属性

computed: {

normalizedFullName: function () {

return this.fullName.split(' ').map(function (word) {

return word[0].toUpperCase() + word.slice(1)

}).join(' ')

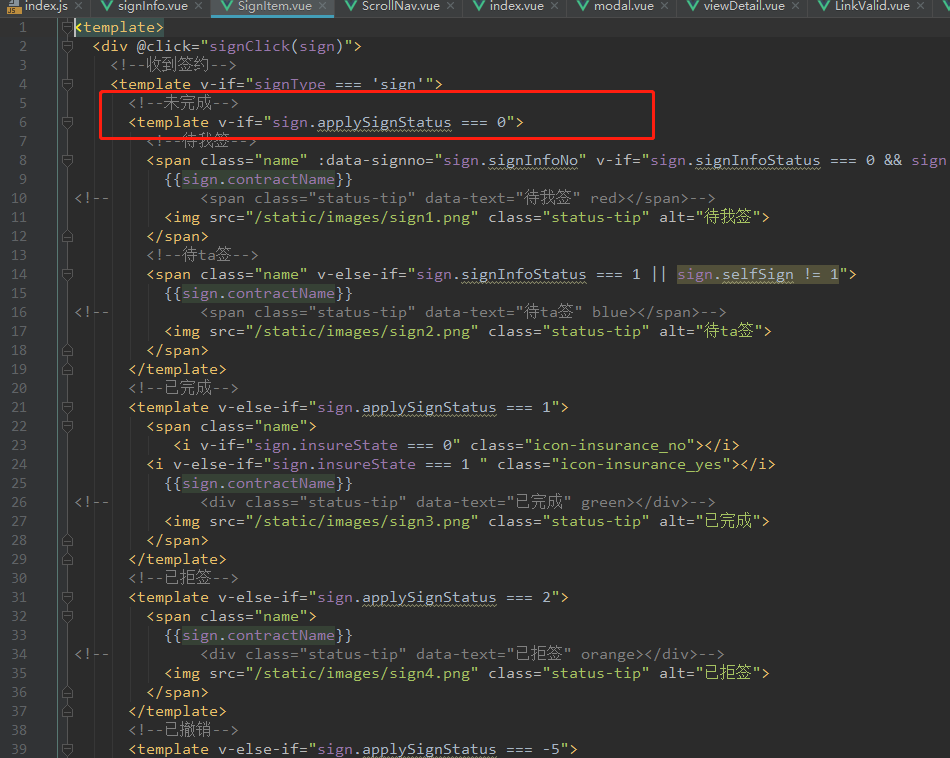
}

}

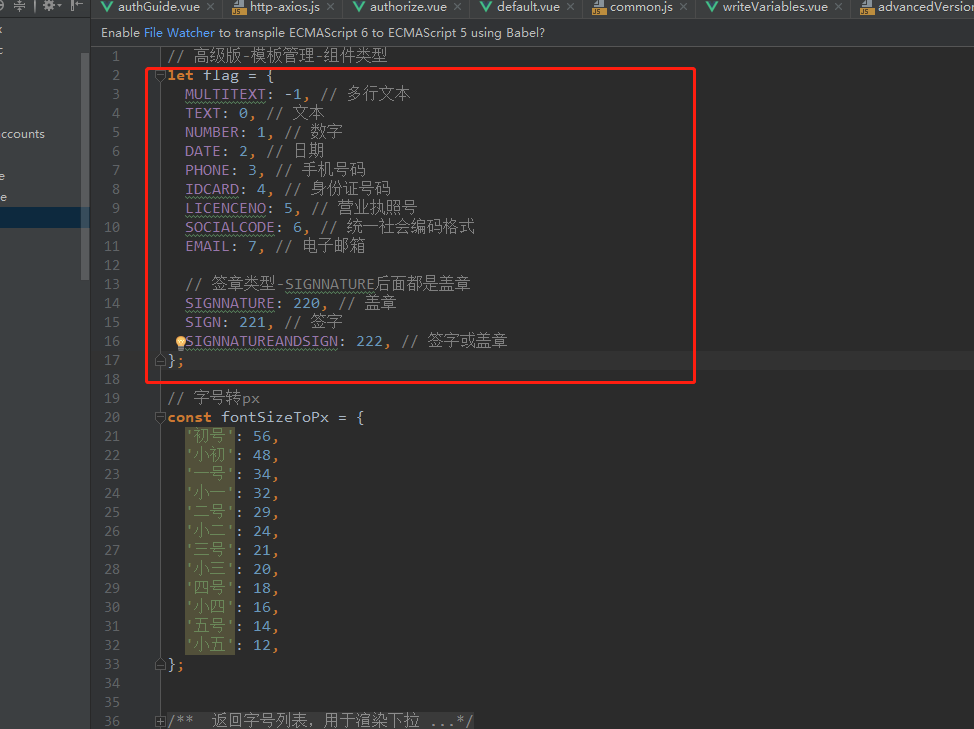
1. Vue 页面排版参考

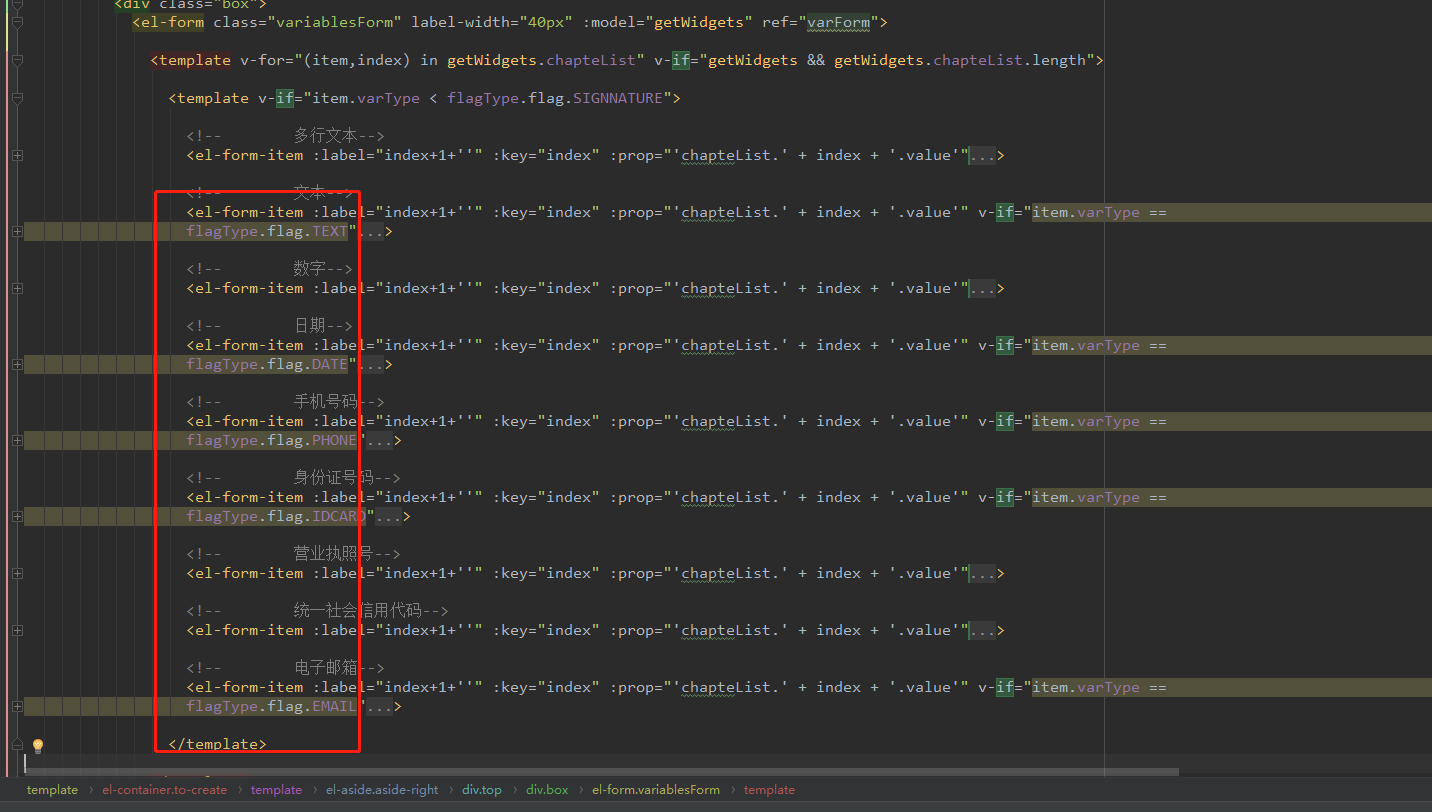


1. if else ，switch等判断写好注释，包括html片段中

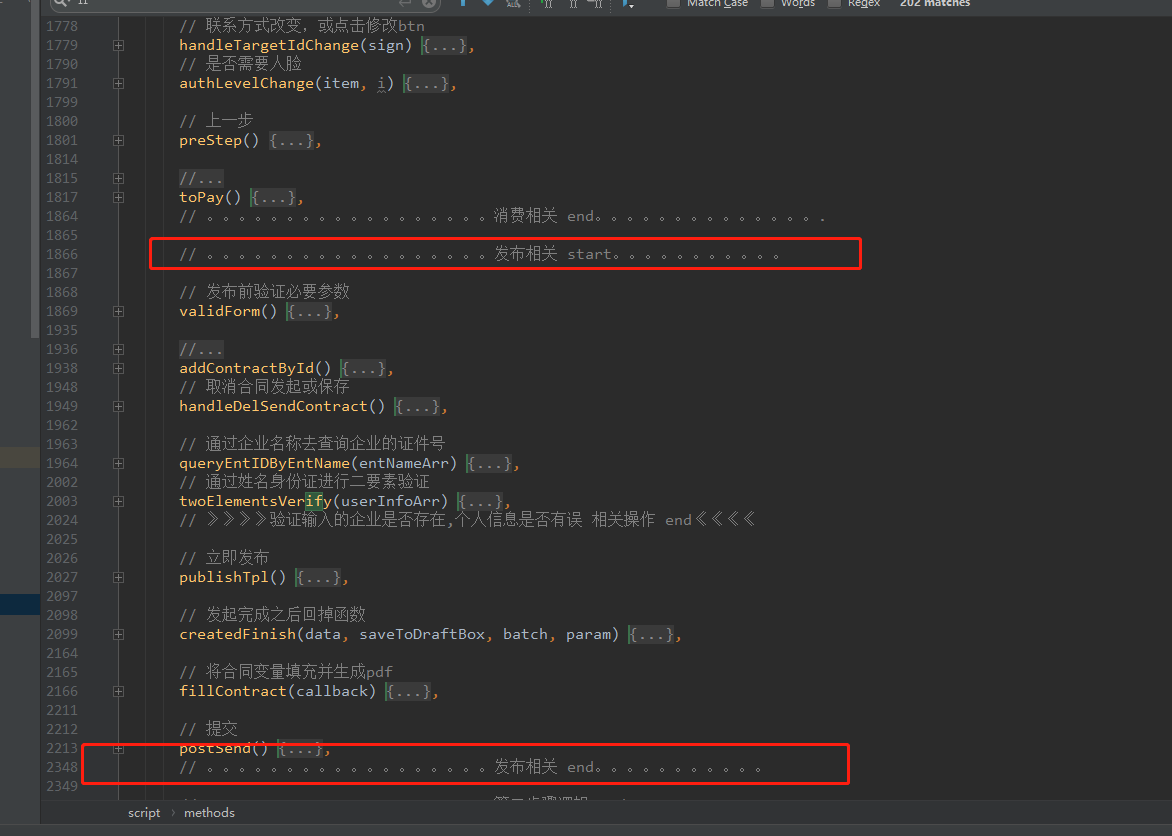


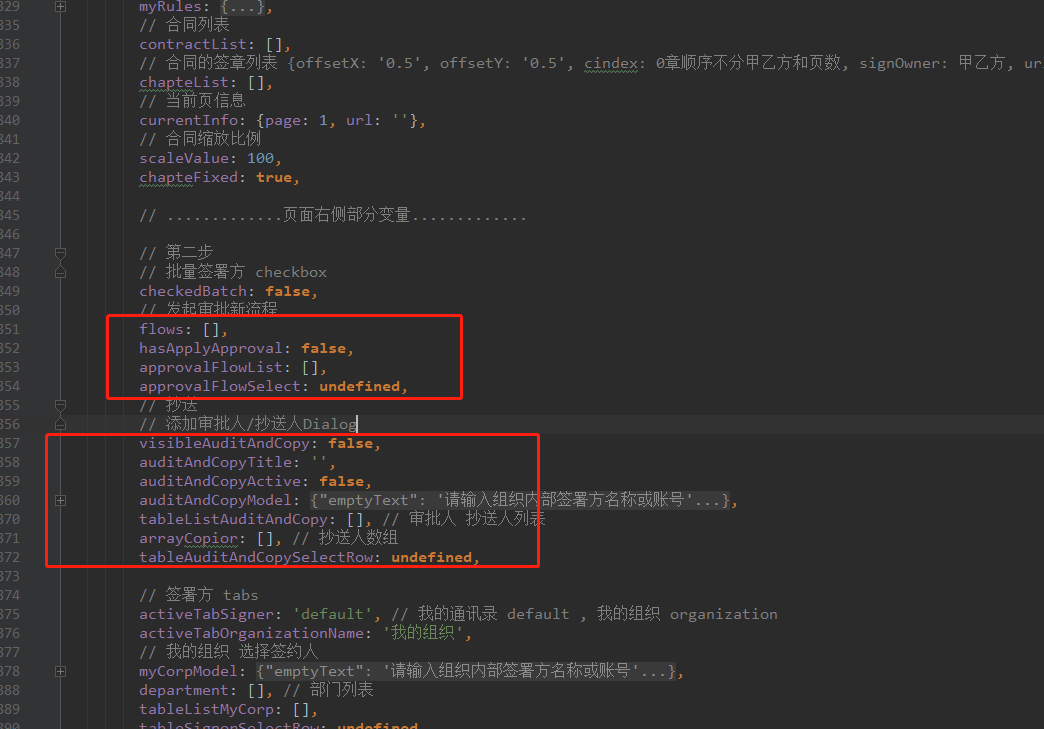
1. 多种类型，多处使用，单独写在data中，或者js中，避免在页面中使用数字直接进行判断，并做好注释





1. 页面复杂时，需要将方法归类并写好注释，同类逻辑操作放一起（便于后续阅读）



21.定义变量时做好分类，相关逻辑的写一起

22.遇到参数过多时，标注好哪些是后端需要的，哪些是前端辅助变量，后续建议这种前端辅助变量的都以 ”\_” 开头，如”\_emailOrPhone”

