# 7f0b84b476eae6707338bdd14aee930

# 与燕科技前端规范化文档demo v1.0 by 赵明章 2019.05

# **update 2019.07**

本文档主要规范的是基于Vue.Js + ElementUI开发的项目框架，项目使用Vue脚手架搭建，因此项目目录沿用脚手架生成的目录， 本文只涉及到基础目录以外的部分。

项目框架基于ElementUI的组件以及部分样式，并作出一些定制化的改动：样式、组件嵌套等。

目录

1. *HTML规范*
2. *CSS规范*
3. *Js规范*
4. *文件目录*
5. *杂项以及配置*

*6.统一页面样式*

## HTML规范

Html部分的代码应保持结构严谨、语义化正确以及良好的视觉感受；

避免标签错误的嵌套，标签以语义优先，不可样式决定标签的使用；

必要时使用注释区分区块等。

#### 1 根元素

根元素是位于template下第一层级元素，只能存在一个，统一使用DIV。

#### 2 元素位置

元素的放置位置也应遵循其在DOM中语义的相同位置，例如：p > span。

如dialog这样的组件，需要放在template的根元素内，并放在最后的位置。

#### 3 栅格

在Element中提供了一套完整的布局layout，应正确地使用它，在涉及到布局的时候优先考虑栅格，例如页面的左右固定布局，表单的布局等，如果只是简单的靠右浮动、居中等，则无须使用栅格。

#### 4 属性绑定

属性绑定时，统一使用 “:” 代替 ”v-bind:”。

#### 5 命名空间

方法的命名使用帕斯卡命名法(驼峰)，方法命名需要可读具有语义， 例如showDialog()， deleteMember()。

#### 6 样式分离

在HTML代码的部分，不应直接出现style属性以及内容（内嵌样式）， 样式统一使用class、id来套用。

#### 7唯一键值

使用v-for时，确保每个元素有独立的key值，不可重复，如遇两处循环渲染都使用同一数据， 在key值定义时需要区分开。

#### 8 注册实例

在Element中有众多可以操作实例的组件，实例在注册时同样按照驼峰命名法命名，并做到具象化，例如：memberInfoTree;

#### 9 ElementUI组件

使用Element组件时，除了依照文档的指引使用，也应具有一定的代码规范，本小结将规定一些常用组件的使规范, 按照字母排序：  
 所有组件有size区分的， 默认使用middle；

El-button 按钮  
 最主要的操作按钮type =”primary”  
 次级操作按钮type =”primary” plain  
 通常按钮 type=”default”

警告按钮 type=”warning”（该操作默认有操作提示， 操作通常是可逆的）

危险按钮 type=”danger”（该操作默认有操作提示，为不可逆操作）

El-dialog 对话框  
 默认添加 :close-on-click-modal=”false”  
 绑定的控制显示变量，统一命名为xxxDialogVisible

El-form 表单  
 通常情况下, 任何表单都需要注册一个ref，单纯只需要展示内容的表单则无需注 册；  
 表单内需要控制行内显示时， 使用栅格控制布局， 栅格放在表单内；  
 不在label后直接使用全角冒号 用label-suffix统一设置

El-tree 树

除了树的基本配置以外，有时候会需要做浮动显示的操作按钮组（text类型）

在src\views\menuExample\functionFolder\functionFolder2\function3 下有示例

#### 10 属性折行

Element组件有时会用到大量的实例注册、属性注册 、方法注册等， 这将导致一行代码过长，所以在遇到属性过多时，应在第一个属性开始前折行书写：

<el-form

ref=”demoForm”

...

:model=”demoForm”>

...

</el-form>

## CSS规范

Css使用less/scss的语法以及结构，框架会提供一些基础的的公用样式， 方便在书写时快速进行样式的细微调整，同时也避免了在HTML中出现style样式属性。

#### 1 对象化

使用less、scss后 样式将使用对象化的风格，即选择器之间存在父子关系，所有选择器应正确的嵌套使用，如此可尽量避免样式穿透、被覆盖等问题。

#### 2 书写顺序

样式应按照一定的书写规范进行：

Position、dispaly、overflow等定位以及相关属性应放在最前；

Margin、padding、height、width等边距和宽高属性次之；

Background、boder、color、font等装饰性样式放在最后。

#### 3 命名空间

Class、id选择器的命名， 使用短横线命名法；

命名时应具有语义，不可直接使用box， block等过于抽象或毫无意义的字母作为名称。

#### 4 作用域

组件内的样式都应加上scoped，避免样式污染。在特殊需求时必须做出穿透修改时，可在另一个style标签中书写，并在本组件的根元素定义好选择器，样式都要写在此选择器下。

#### 5 权重

当样式权重不足时，尽量避免出现!important， 使用多个选择器来增加权重；

如果遇到样式权重实在无法覆盖时，则可以使用!important，但要确保不要污染到其他同种元素或组件的样式；

## JS规范

使用ES6的语法， 保持代码简洁易读， 保持良好的注释书写习惯；

Vue项目的js代码风格是面向对象风格，所有的组件中用到的js都是一个对象，对象中存在data，methods属性等。

#### 1 初始化

Vue是一个数据驱动的框架，数据是项目关键的一环， 所有的变量都应提前在data中提前创建储存空间，并规定好变量类型，如果初始值无需特殊定义，通常情况下定义为 null（特殊情况下初始值可以定义为undefined）；

#### 2 命名空间

变量以及方法的命名都应遵循匈牙利命名法和驼峰命名法，不可以用简略字母或数字代替， 例如：memberInfos， buttonClicked， handleTypeChanged等；

在互相理解的情况下可以适当使用简写，例如：num，err等；

函数的参数params可以抽象命名，无需具体到特例，但如果在HTML中绑定events所传递的参数名，应在methods中与其保持一致；

注册在Vue对象的全局对象 以$符开头

#### 3 书写顺序

遵循vue的生命周期，属性按照如下顺序书写（多行属性定义时，按照如下方式对齐）

Components 组件

Props 属性

Data 数据

Computed 计算属性

Watch 监听

Created 生命周期-创建

Mounted 生命周期-完成

Methods 方法  
 methods中的方法也应该按照一定的循序书写，可以按照页面触发的位置来书写， 例如：顶部的相关函数放在最前，分页相关的方法函数放在最后。

#### 4 请求

请求统一使用axios；

在处理内容少的情况，可将逻辑直接写在回调中，逻辑较复杂时，应提为一个单独的method，放在次请求的后方；

链式请求时，使用Promise 或者 async/await，二者优缺点都有，依据情况使用；

#### 4 全局变量

当全局变量被注册时，在变量名前加上$符

在vuex中注册的变量也是全局的，但不需要添加$符

#### 5 组件

每一个页面组件都拥有一个独有的name，不可重复，name需要在每个vue组件中声明， 并且与其在router中注册的name一致

## 文件目录

在基于脚手架的基础上，后续新创建的文件以及文件结构也存在一定的规范内容。

所有页面文件统一存放在view文件夹下，每一个大类应创建一个文件夹存放。

#### 1 命名空间

新创建的文件和文件夹都应使用驼峰命名法则，新的vue文件（组件）使用帕斯卡命名法，即大驼峰；

#### 2 组件

当前xxx.vue页面使用了外部组件，应在当前目录中创建一个xxxComponents文件夹用于存放这个页面所用到的组件。

#### 3嵌套路由

当页面存在嵌套路由的情况时，需要在children中注册，并用一个Index.vue存放router-view， 然后将其余页面和Index.vue一起放入新建的文件夹中（原理可参考vue-router文档）  
 路由的嵌套规则应和文件夹对应的嵌套关系一致。  
 列表页和详情页无需分层级, 在同一目录下配置redirectName(列表的Name)  
system(Folder)

┝ Index.vue

┕ personManage(Folder)

┝ Index.vue

┝ List.vue

┕ Detail.vue  
↓对应路由↓

router: [

{

systemIndex,

children: [

{

personManageIndex,

children: [

List,

Detail

]

}

]

}

]

## 杂项以及配置

#### 1 制表符

使用空格代替制表符， 同时制表符长度为2个空格

#### 2 代码块

当一部分代码行数过多并且具有区域性或分步性时， 应将同类型的代码放在相邻的位置，必要时不同的区块应用一个空白行或者注释行分隔。例如

let xxx = 1

xxx.push(...)

let yyy = {}  
 yyy...

#### 3 Router

路由元需要携带参数以及配置项时，通常写在meta属性中  
 但如果有特殊需要也可以写在meta同级， 但要注意命名不要占用关键字  
 多级路由，每一级都需要有一个router-view组件用以渲染

Ruoter的meta属性可选值在router.js下有详细说明

## 统一页面样式

为方便管理以及样式统一，页面有一套固定的基础样式体系，所有页面（除开登录页，404页等）都遵循这个样式体系。

详细可以参考/view/menuExample下的文件样式