**命名规范**

1. 文件夹命名规范

* **命名方式：Camel Case 小驼峰式命名法：首字母小写。eg：studentInfo、userInfo、productInfo**
* **命名规范：根据功能模块划分**
  + demo：
  + order 订单模块
  + userInfo 用户模块

2. 文件命名规范

* **命名方式：Pascal Case 大驼峰式命名法：首字母大写。eg：StudentInfo、UserInfo、ProductInfo**
* **命名规范：根据功能模块划分**
  + demo：
  + HistoryOrder 历史订单

3. 变量命名规范

* **命名方式：Camel Case 小驼峰式命名法：首字母小写。eg：studentInfo、userInfo、productInfo**
* **命名规范：类型+对象描述的方式，如果没有明确的类型，就可以使前缀为名词**

4. 常量

* **命名方法:全部大写**
* **命名规范：使用大写字母和下划线来组合命名，下划线用以分割单词**

5. 函数命名规范

* **命名方式：小驼峰方式 ( 构造函数使用大驼峰命名法 )**
* **命名规则：前缀为动词**
  + 动词 含义 返回值
  + can 判断是否可执行某个动作 ( 权限 ) 函数返回一个布尔值。true：可执行；false：不可执行
  + has 判断是否含有某个值 函数返回一个布尔值。true：含有此值；false：不含有此值
  + is 判断是否为某个值 函数返回一个布尔值。true：为某个值；false：不为某个值
  + get 获取某个值 函数返回一个非布尔值
  + set 设置某个值 无返回值、返回是否设置成功或者返回链式对象
  + demo：
  + //是否可阅读 function canRead(){ return true; } //获取姓名 function getName{ return this.name }

**vue组件的开发规范**

1. 依照vue生命周期顺序放置对应的方法

1. components
2. props
3. data
4. computed
5. filter
6. metods
7. created
8. mounted
9. activited
10. update
11. beforeRouteUpdate
12. watch

2. 父子组件传参禁止子组件去修改父组件的参数

* 通过 props 将值传递给子组件
* 通过 props 传递回调函数给子组件来达到调用父组件方法的目的
* 通过在子组件触发事件来通知父组件

3. vue开发文件夹规范

src assets ... components 基础组件 ... plugs 基础插件 ... views comms 多组建业务复用提取到comms Header.vue Footer.vue car CarEdit.vue CarList.vue CarDetai.vue user UserDetail.vue UserEdit.vue UserPasswordRest.vue customer CustomerCardItem.vue CustomerList.vue

每个目录作为一个页面层级，可以包含子页面，且必须存在一个index.js

4. vue文件加上name属性，方便拓展与devtools 测试

5. 使用组件名作为样式作用域空间

* 给样式加上作用域空间可以避免组件样式影响外部的样式
* 保持模块名、目录名、样式根作用域名一样，可以很好的将其关联起来，便于开发者理解。

6. 组件表达式简单化

* 复杂的行内表达式难以阅读。
* 行内表达式是不能够通用的，这可能会导致重复编码的问题。
* IDE 基本上不能识别行内表达式语法，所以使用行内表达式 IDE 不能提供自动补全和语法校验功能。
* 如果写了太多复杂并难以阅读的行内表达式，使用 method 或是 computed 属性来替代其功能。

7. 子组件props规范

* 提供默认值
* 使用 type 属性 校验类型
* 使用 props 之前先检查该 prop 是否存在

8. 组件的name使用大写开头的驼峰法，对应到组件标签时驼峰演变成全小写横线连接

* name: UserDetail.vue
* 引用组件{user-detail}

**vuex的使用**

1. 禁止在除了mutations之外的任何地方去修改store

2. 引入组件的vuex只能mapGetters以及mapActions，禁止直接操作store

3. 强制开启namespace和strict