# 大[前端规范总结](https://www.cnblogs.com/janessasmith/p/4500999.html)

## **一、概述**

本规范旨在为前端程序的开发者提供规范化最新的指导，可用于程序员个人编译环境以及研发团队集成环境等场合的代码规范化检查。

## **二、方针**

| **约束等级** | **约束效力** | **强制性** |
| --- | --- | --- |
| 【强制】 | 违反该项将被认为代码存在严重缺陷 | 前端程序团队必须遵守 |
| 【推荐】 | 违反该项将被认为代码存在轻微缺陷 | 根据具体产品特性的不同，选择性地遵守 |
| 【参考】 | 违反该项可被认为代码存在优化空间 | 从产品持续优化及人员技能提升的角度，参考使用 |

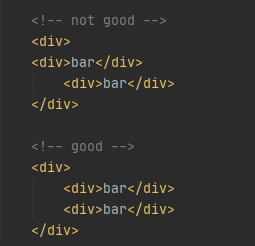
## **三、环境要求**

1. 【强制】[Node.js](https://nodejs.org/) 10.x 或更高版本，你可以使用 [nvm](https://github.com/creationix/nvm) 或 [nvm-windows](https://github.com/coreybutler/nvm-windows)在一台电脑中管理多个 Node 版本
2. 【强制】使用 [WebStrom](http://www.baidu.com/link?url=Sn0aBa011utoLAP3IC799QxEcwe1-mmChrkEchQsAaOJJwwbN8ZMTyf8XF8uDId9)或 [Visual Studio Code (VS Code)](http://www.baidu.com/link?url=Sn0aBa011utoLAP3IC799QxEcwe1-mmChrkEchQsAaOJJwwbN8ZMTyf8XF8uDId9) 进行代码编写,前端开发时不接受使用其他开发工具。如使用其他语言开发时可使用想对应的开发工具,例：IntelliJ IDEA，eclipse，pyCharm等
3. 【强制】代码提交前进行代码格式化（不要格式引入的外部文件）
4. 【推荐】使用 Chrome 浏览器并安装 [Vue.js devtools](https://chrome.google.com/webstore/detail/vuejs-devtools/nhdogjmejiglipccpnnnanhbledajbpd) 进行调试

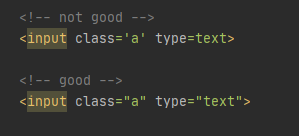
## **四、编码规范**

### 4.1 HTML / Template 编码规范

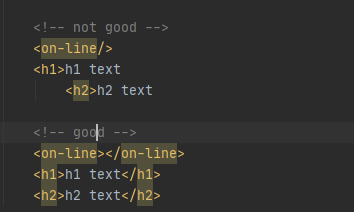
1. 【强制】嵌套元素应当缩进一次，同层级缩进应保持一致



1. 【强制】对于属性的定义，使用双引号，不要使用单引号



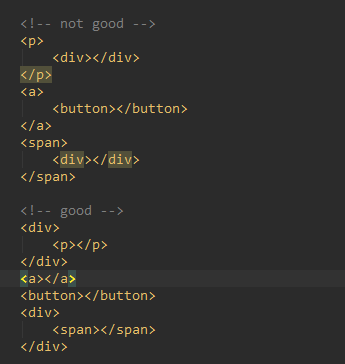
1. 【强制】不要省略可选的闭合标签,自定义组件使用闭合标签。(省略可选的结束标签，虽不会违反 H5 规范；但可能会造成层级上的困扰，导致代码出现无法预料的问题)



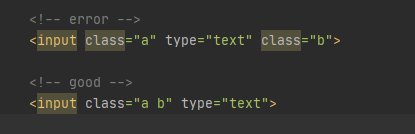
1. 【强制】特殊符号使用 [HTML 字符实体](http://www.w3school.com.cn/html/html_entities.asp)（实体名称对大小写敏感），常用如下：

| **符号** | **实体编码** |
| --- | --- |
| 空格 | &nbsp; |
| © | &copy; |
| ¥ | &yen; |
| ® | &reg; |
| > | &gt; |
| < | &lt; |
| & | &amp; |

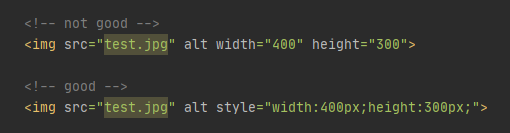
1. 【强制】语义嵌套约束，严格嵌套约束



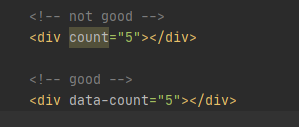
1. 【强制】不使用重复属性



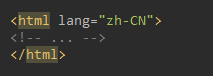
1. 【推荐】不使用属性设置样式（img, table等元素）



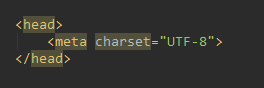
1. 【参考】自定义属性要以data-开头 自己添加的非标准的属性要以data-开头，否则[w3c validator](https://link.juejin.im/?target=https://validator.w3.org/)会认为是不规范的



1. 【强制】为每个 HTML 页面根元素添加 [lang 属性](http://w3c.github.io/html/semantics.html" \l "the-html-element)(强烈建议为 html 根元素指定 lang 属性，从而为文档设置正确的语言。这将有助于语音合成工具确定其所应该采用的发音，有助于翻译工具确定其翻译时所应遵守的规则等等)



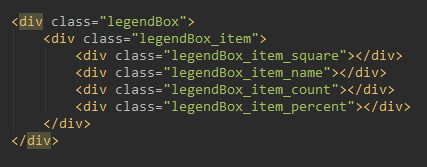
1. 【强制】通过声明一个明确的字符编码，让浏览器轻松、快速的确定网页内容渲染方式，通常指定为'UTF-8'



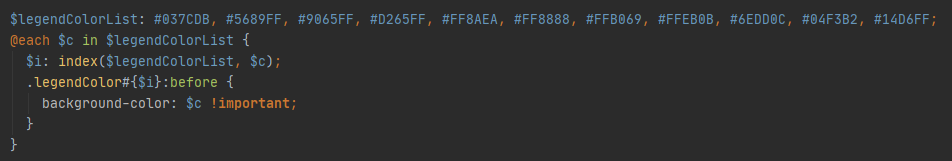
1. 【参考】属性应该按照特定的顺序出现以保证易读性
2. class
3. id
4. name
5. data-\*
6. src, for, type, href, value , max-length, max, min, pattern
7. placeholder, title, alt
8. aria-\*, role
9. required, readonly, disabled
10. 【参考】尽量遵循 HTML 标准和语义，但是不要以牺牲实用性为代价；任何时候都要尽量使用最少的标签并保持最小的复杂度。

### **4.2 CSS / Scss 编码规范**

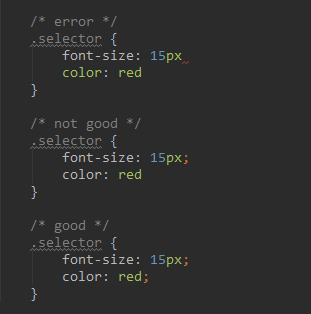
1. 【强制】类名使用小驼峰命名，以下划线分隔



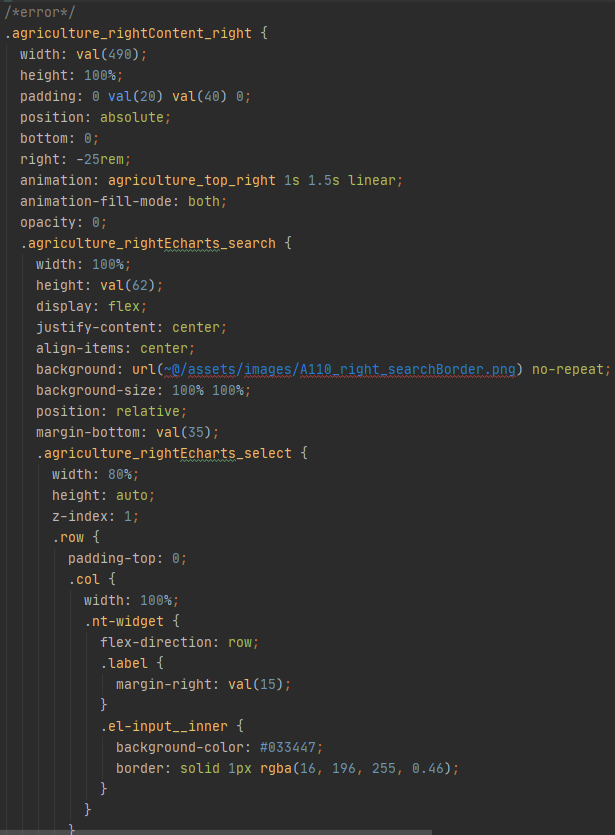
1. 【强制】Scss中的变量、函数、混合等采用驼峰式命名



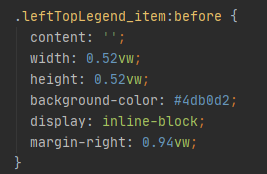
1. 【强制】所有声明语句都应当以分号结尾 最后一条声明语句后面的分号是可选的，但是，如果省略这个分号，你的代码可能更易出错



1. 【强制】避免为 0 值指定单位，例如，用 margin: 0; 代替 margin: 0px;
2. 【强制】十六进制值应该全部小写，例如，#f3f6fa
3. 【强制】多写注释，且多使用句子进行描述而不是词语
4. 【强制】选择器不要超过3层（在 Scss 中避免嵌套超过 3 层）



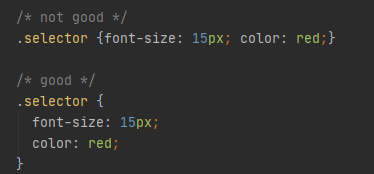
1. 【推荐】不推荐使用\*选择器
2. 【推荐】适当使用:before和:after来画页面的一些视觉上的辅助性元素，如三角形、短的分隔线、短竖线等，可以减少页面上没有用的标签



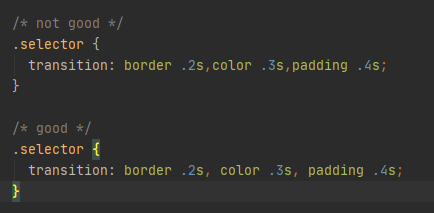
1. 【推荐】使用简写形式的十六进制值，例如，用 #fff 代替 #ffffff

##### 以下大部分工作将由代码格式化工具完成（参见环境要求），一般无需考虑

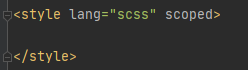
1. 【强制】声明块的左花括号前添加一个空格
2. 【强制】声明块的右花括号应当单独成行
3. 【强制】每条声明语句的 : 后应该插入一个空格
4. 【强制】每条样式声明应该独占一行



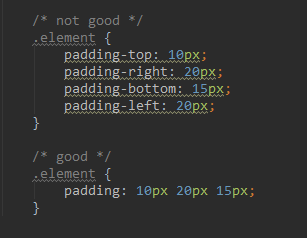
1. 【强制】对于以逗号分隔的属性值，每个逗号后面都应该插入一个空格（例如，box-shadow，transition）



1. 【强制】!important前插入一个空格
2. 【强制】注释：//后插入一个空格，/\*后插入一个空格，\*/前插入一个空格
3. 【推荐】单行注释用// 多行注释统一用/\* \*/
4. 【强制】非通用样式使用嵌套方式进行编写，避免影响其他自己不了解样式，造成样式覆盖，vue页面中使用作用域来避免样式污染



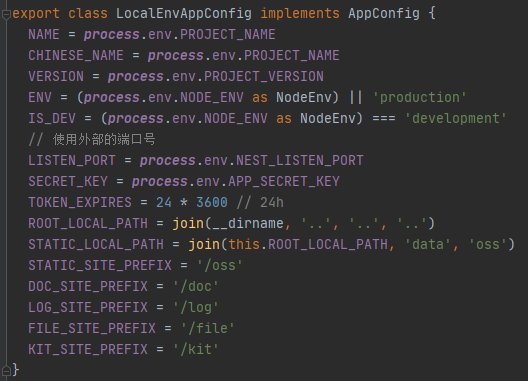
1. 【强制】不使用!important，除非原样式使用内联样式或!important且无法直接修改
2. 【强制】无法修改原样式声明时，应通过权重关系，编写权重更高的样式进行覆盖
3. 【推荐】样式需要修改时，尽量找到原样式声明进行修改
4. 【推荐】当你确定自己的声明不会影响到其他属性时，请使用简写提升代码简洁性



### **4.3 JavaScript 编码规范**

#### **命名**

1. 【强制】标准变量采用驼峰式命名（考虑与后台交换数据的情况，对象属性可灵活命名）
2. 【强制】常量全大写，用下划线连接



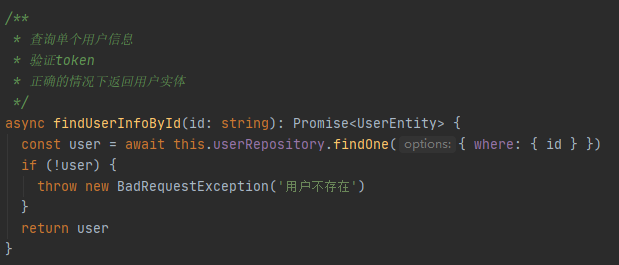
1. 【强制】变量名不应过短，要能准确完整地描述该变量所表述的事物

| **不好的变量名** | **好的变量名** |
| --- | --- |
| inp | input, priceInput |
| day1, day2, param1 | today, tomorrow |
| id | userId, orderId |
| obj | orderData, houseInfos |
| tId | removeMsgTimerId |
| handler | submitHandler, searchHandler |

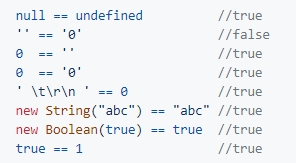
1. 【强制】变量名不要使用计算机术语，如 texareaData，应该取和业务相关的名字，如 leaveMsg
2. 【强制】变量名的对仗要明确，如 up/down、begin/end、opened/closed、visible/invisible、scource/target
3. 【强制】变量名使用正确的语法(不要使用中文拼音，如 shijianchuo 应改成 timestamp ； 如果是复数的话加 s，或者加上 List，如 orderList、menuItems； 而过去式的加上 ed，如 updated/found 等； 如果正在进行的加上 ing，如 calling；)
4. 【推荐】使用临时变量时请结合实际需要进行变量命名
5. 【推荐】布尔变量名应使用肯定的布尔变量名，不要使用否定的名词，如 notOk、notReady，因为否定的词取反的时候就会比较奇怪，如 if (!notOk)

#### **语法**

1. 【强制】变量不要先使用后声明
2. 【强制】不要声明了变量却不使用
3. 【强制】不要在同个作用域下声明同名变量
4. 【强制】一个函数作用域中所有的变量声明尽量提到函数首部，可根据代码进行分组
5. 【强制】为了快速知晓变量类型，声明变量时要赋值(也可以在注释中说明类型)
6. 【强制】单一函数的返回值类型要确定



1. 【强制】debugger不要出现在提交的代码里
2. 【推荐】使用===代替==，!==代替!=（==会自动进行类型转换，可能会出现奇怪的结果）



1. 【推荐】使用三目运算和逻辑计算符代替简单的 if-else



1. 【推荐】使用let定义变量，const定义常量
2. 【推荐】将复杂的函数分解成多个子函数，方便维护和复用

#### **代码风格**

1. 【强制】对象属性名不需要加引号
2. 【强制】对象以缩进的形式书写，不要写在一行
3. 【强制】数组中不要存在空元素

#### **使用 null**

1. 【强制】正确使用 null

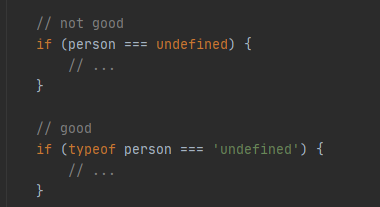
适用场景：

* 初始化一个将来可能被赋值为对象的变量
* 与已经初始化的变量做比较
* 作为一个参数为对象的函数的调用传参
* 作为一个返回对象的函数的返回值

1. 不要用null来判断函数调用时有无传参
2. 不要与未初始化的变量做比较

#### **使用 undefined**

1. 【强制】正确使用 undefined
2. 不要给变量赋值 undefined（undefined 本身就表示一个变量未定义）
3. 不要直接使用 undefined 进行变量判断
4. 使用typeof和字符串 'undefined' 对变量进行判断



#### **文档注释**

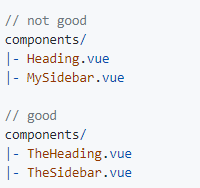
1. 【参考】各类标签 @param, @method 等请参考 [usejsdoc](http://usejsdoc.org/)和 [JSDoc Guide](http://yuri4ever.github.io/jsdoc/)；

### **4.4 Vue 组件编码规范**

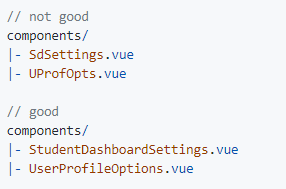
1. 【强制】根组件名应该始终文件名
2. 【强制】单文件组件的文件名遵循小驼峰命名
3. 【强制】和父组件紧密耦合的子组件应该以父组件名作为前缀命名
4. 【推荐】应用特定样式和约定的基础组件 (也就是展示类的、无逻辑的或无状态的组件) 应该全部以一个特定的前缀开头，比如 Base、App



1. 【推荐】只应该拥有单个活跃实例的单例组件应该以 The 前缀命名，以示其唯一性



1. 【推荐】组件名应该倾向于完整单词而不是缩写



1. 【强制】组件引入时如果超出一级，使用别名引入



1. 【强制】prop 的定义应该尽量详细，至少需要指定其类型



1. 【强制】为 v-for 设置键值；在组件上总是必须用 key 配合 v-for，以便维护内部组件及其子树的状态
2. 【强制】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
3. 【强制】可简写指令需要缩写 (用 : 表示 v-bind: 和用 @ 表示 v-on:)
4. 【推荐】不要把 v-if 和 v-for 同时用在同一个元素上(vue3.0 v-if 和v-for的优先级改了)
5. 【推荐】应该把复杂计算属性分割为尽可能多的更简单的属性

#### **组件/实例的选项的顺序**

【参考】组件/实例的选项应该有统一的顺序，这是推荐的组件选项默认顺序：

1. ****副作用**** (触发组件外的影响)
   1. el
2. ****全局感知**** (要求组件以外的知识)
   1. name
   2. parent
3. ****组件类型**** (更改组件的类型)
   1. functional
4. ****模板修改器**** (改变模板的编译方式)
   1. delimiters
   2. comments
5. ****模板依赖**** (模板内使用的资源)
   1. components
   2. directives
   3. filters
6. ****组合**** (向选项里合并属性)
   1. extends
   2. mixins
7. ****接口**** (组件的接口)
   1. inheritAttrs
   2. model
   3. props/propsData
8. ****本地状态**** (本地的响应式属性)
   1. data
   2. computed
9. ****事件**** (通过响应式事件触发的回调)
   1. watch
   2. 生命周期钩子 (按照它们被调用的顺序)
      1. beforeCreate
      2. created
      3. beforeMount
      4. mounted
      5. beforeUpdate
      6. updated
      7. activated
      8. deactivated
      9. beforeDestroy
      10. destroyed
10. ****非响应式的属性**** (不依赖响应系统的实例属性)
    1. methods
11. ****渲染**** (组件输出的声明式描述)
    1. template/render
    2. renderError

#### **元素特性的顺序**

【参考】元素 (包括组件) 的特性应该有统一的顺序，这是我们为元素特性推荐的默认顺序：

1. ****定义**** (提供组件的选项)
   1. is
2. ****列表渲染**** (创建多个变化的相同元素)
   1. v-for
3. ****条件渲染**** (元素是否渲染/显示)
   1. v-if
   2. v-else-if
   3. v-else
   4. v-show
   5. v-cloak
4. ****渲染方式**** (改变元素的渲染方式)
   1. v-pre
   2. v-once
5. ****全局感知**** (需要超越组件的知识)
   1. id
6. ****唯一的特性**** (需要唯一值的特性)
   1. ref
   2. key
   3. slot
7. ****双向绑定**** (把绑定和事件结合起来)
   1. v-model
8. ****其它特性**** (所有普通的绑定或未绑定的特性)
9. ****事件**** (组件事件监听器)
   1. v-on
10. ****内容**** (覆写元素的内容)
    1. v-html
    2. v-text

#### **单文件组件的顶级元素的顺序**

【强制】单文件组件应该总是按照 <template>、<script> 和 <style> 的标签顺序

#### **隐性的父子组件通信**

【强制】应该优先通过 prop 和事件进行父子组件之间的通信，而不是 this.$parent 或改变 prop。

#### **全局状态管理**

【强制】应该优先通过 Vuex 管理全局状态，而不是通过 this.$root 或一个全局事件总线。

## **组件复用**

【推荐】避免重复造轮子，多使用成熟的现成工具/类库/组件，如：lodash、qs、url-parse、date-fns/format 等

模块设计原则：

* 高内聚低耦合、可扩展
* 不要去改变模块输入的数据 (引用类型)，如：函数参数、组件 prop
* 模块的入参为可选 Boolean 时，默认值应设计为 false

【强制】严格遵守 ESLint 语法校验，警告级别的也要处理 (暂时用不到的代码可以先注释掉)

## **给 UI 的建议**

在画 UI 界面时，建议参考前端已选型的开源组件库，并推荐使用开源组件库提供的制图元件/模板，如：[element-ui](http://element-cn.eleme.io/" \l "/zh-CN/resource)[, ant-design](https://www.antdv.com/docs/vue/getting-started-cn/)

移动端项目，如：[vant-ui](https://youzan.github.io/vant/" \l "/zh-CN/design)

图表组件，如：[ehcarts](https://www.makeapie.com/explore.html), [amcharts4](https://www.amcharts.com/demos/), [highcharts](https://www.highcharts.com.cn/demo/highcharts)