

代码规范



课程名称 软件工程

题目名称 代码规范报告

任课教师 杨 枨

组 号： G18

组 长： 谢子文

组 员： 梁泽生

组 员： 黄 馨

2020 年12月 9 日

**目录**

[一、 引言： 3](#_Toc31209)

[1.1 规范目的： 3](#_Toc30368)

[1.2 参考资料 3](#_Toc18132)

[1.3 版本历史 3](#_Toc2656)

[二、 规范说明： 4](#_Toc19891)

[三、 命名规范： 4](#_Toc6785)

[3.1. 基本规则： 4](#_Toc28085)

[页面page命名全是小写英文单词组合。 4](#_Toc3445)

[3.1.1. 规则1 4](#_Toc20366)

[3.1.2. 规则2 4](#_Toc7396)

[3.1.3. 规则3 4](#_Toc21230)

[3.1.4. 规则4 5](#_Toc3039)

[3.1.5. 规则5 5](#_Toc28755)

[3.1.6. 规则6 6](#_Toc29648)

[四、 编码规范： 6](#_Toc31422)

[4.1 WXML规范： 6](#_Toc1518)

[4.2 CSS规范： 7](#_Toc18401)

[4.3 JS规范 9](#_Toc19764)

[4.4 组件规范： 10](#_Toc15016)

[五、 注释规范： 11](#_Toc7471)

[5.1. WXSS注释规范 11](#_Toc19300)

[5.2. WXML注释规范 12](#_Toc31774)

[代码中基本上每个模块都要注释一下，以方便浏览者了解这部分代码是干什么用的，WXML里面的注释模式是<!-- -->。 12](#_Toc16850)

[六、 代码走查： 12](#_Toc2877)

# 引言：

## 1.1 规范目的：

1.增强代码可维护性。代码的编写不是一次性就能写得很完美的，需要不断的修复bug，修改或增加功能，重新设计整体架构等。这时就需要进入代码中去做修改，如果没有良好的代码规范，时间久了自己阅读起来就很费力。

2.提高团队开发效率。大多数项目的代码都不是由一个人编写的，其他成员也许会因为项目的交接需要接手管理你所编写的代码，如果没有良好的代码规范，他人便无法快速轻松的理解你的代码。

3.提高个人编码效率。最开始可能会觉得被规范约束，后来反而会因为有规范而有依靠感，不必再为起什么名字而犹豫。

## 1.2 参考资料

[1] 《GB856T——88》国标文档

[2] 《软件工程导论》清华大学出版社，张海藩、牟永敏编著

[3] 代码规范

<https://www.jianshu.com/p/d7e87107073c>

## 1.3 版本历史

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 版本修订日期 | 版本修改人 | 是否通过审核 |
| 代码规范v0.1.0 | 2020/12/28 | 黄馨 | 是 |
| 代码规范v0.1.1 | 2020/12/29 | 黄馨 | 是 |

# 规范说明：

代码的规范不是绝对的，每个公司都会有自己的一套代码规范，每个新人刚进入都要最先学习代码规范，才能加入到团队的开发中来。

# 命名规范：

## 基本规则：

## 页面page命名全是小写英文单词组合。

### 规则1

英文单词命名。禁止使用拼音或无意义的字母命名。

### 规则2

直观易懂。使用能够描述其功能或意义的英文单词或词组。

### 规则3

变量名称命名都是英文的，我们采用小驼峰命名法和下划线命名法结合来给变量命名。正常变量采用小驼峰命名法（对象描述的英文），button类变量会有点小不同，一般是 变量名 = “btn” + 对象描述。例如：



由图可知，class类型的变量命名都为小驼峰命名法:

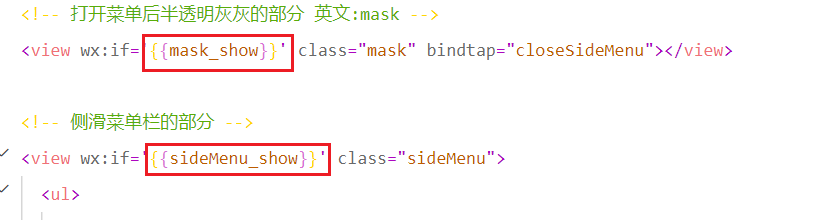
1. Class=“mask”,表示命名这个view名字为mask事件。
2. Class=“sideMenu”，意思就是侧面菜单。



“btn\_showSideMenu”这是一个按钮组件，按下这个按钮，侧面菜单就会滑出来。

### 规则4

一般都是 变量名 = 对象描述 + “\_” + 布局类型；



如图，第一部分中mask是对对象的描述，show是指这是一个展示；第二部分中，sideMenu是对对象的描述，意思是侧边菜单，show指这是一个展示。

### 规则5

部分变量命名是 变量名 = 模块名称 + “\_” + 对象描述，后面的对象描述采用小驼峰式命名法。



例如图中的view\_head、view\_name，它们的模块格式是view类型的，所以就前面是view，后面是描述它们的名称，分别是头像和名字。

### 规则6

还有部分采用的是短横线命名法，变量名 = 对象 + “+” + 对象描述



# 编码规范：

## 4.1 WXML规范：

1. wxml标签可以单独出现的情况，尽量单独出现，如<input />。
2. 每行HTML代码写完整了再换行，看着比较整洁，方便阅读浏览，例如下图：



1. 合理展现分离内容，少使用内联样式。（内联样式直接在标签内部引入，显著的优点是十分的便捷、高效；但是同时也造成了不能够重用样式的缺点，如果代码行数到达一定长度不建议采用内联样式。）

例如这就是内联样式：

      <view class="view\_interface common" style="color:{{style\_fontColor}}">

        开发者界面

      </view>

## 4.2 CSS规范：

1. 在开发过程中rpx和px均可能用到，如通常情况下间距使用rpx，模块边角光滑度和模块阴影使用px，开发者根据实际情况而定。

例如：

使用rpx:  width: 250rpx;

height: 200rpx;

使用px:  border-radius: 10px;

box-shadow:0px 10px 30px rgba(0, 0, 0, 0.24);

1. CSS代码需有明显的代码缩进。每一个样式类之间空出一行。

例如：

.goods-name{

  width: 250rpx;

  text-align: center;

  margin: 0;

}

.iconPrice{

  display: flex;

  flex-flow: row nowrap;

  justify-content: center;

  padding: 25rpx;

}

1. 尽量使用简写属性，并且同一模块的属性放置在一起，避免散乱。

下例中，第一个属于个人中心模块，个人中心里面包含了用户头像，用户名字，用户的userid，还有一个箭头图片布局。模块以内的东西都放置在一起。

.view\_personalCenter {

  width: 95%;

  height: 100px;

  background-color: #FFFFFF;

  margin: 5px auto;

  border: 6px solid #F5F5F5;

  border-radius: 15px;

}

.view\_head {

  overflow: hidden;

  border-radius: 50%;

  width: 70px;

  height: 70px;

  margin: 30rpx 60rpx;

}

.view\_name {

  line-height: 6rpx;

  width: 100px;

  height: 20px;

  margin-top: 35px;

  font-size: 15px;

}

.view\_uid {

  width: 70px;

  height: 20px;

  margin-top: 25px;

  font-size: 15px;

}

.view\_arrow {

  width: 20px;

  height: 20px;

  display: inline-block;

  margin-right: 5px;

}

1. 采用flex进行布局，禁止使用float以及vertical-align。

.container{

disaplay: flex;

flex-dirextion: row

}

## 4.3 JS规范

1.命名规范：变量名以及函数名一般采用小驼峰命名法，偶尔采用小驼峰命名法与下划线命名法结合，两种命名法结合的情况一般是变量属性加描述类型。常量需用const 声明。类的命名首字母需大写。采用ES6 关键字let定义变量，尽量不使用var。

定义常量：const a=1

定义变量：

    /\* swiper左右滑动模块需要用到的data \*/

    currentTab:0,

    indicatorDots: true,

    vertical: false,

    autoplay: false,

    circular: false,

    interval: 2000,

    duration: 500,

    previousMargin: 0,

    nextMargin: 0,

1. 设置数组类的变量规范，不同类型的数组数据格式也不太一样。

如果每个数组内都有点不同布局，就需要以下样式：

    sheetArray:[

      {recordNum:128, name:"用户信息"},

      {recordNum:128, name:"蒸品种类"},

      {recordNum:128, name:"xx信息"},

      {recordNum:128, name:"xx信息"},

      {recordNum:128, name:"xx信息"},

      {recordNum:128, name:"xx信息"},

      {recordNum:128, name:"xx信息"},

      {recordNum:128, name:"xx信息"},

      {recordNum:128, name:"xx信息"},

    ],

    ifClass:['userTBL','steamGoodsTBL'] // 类名想同直接添加

  },

如果是一致的数字类数组，直接按顺序填写数字就行了。

    /\* 设置时间的data \*/

    timesetArray: [ //picker选择时间用的数组

      '10:00', '15:00', '20:00', '25:00', '30:00', '35:00', '40:00', '45:00',

      '50:00', '55:00', '60:00',  '65:00', '70:00', '75:00', '80:00', '85:00',

      '90:00', '95:00', '100:00', '105:00',  '110:00', '115:00', '120:00'

    ],

    timesetArray\_num: [ //timesetArray对应的number类型的分钟数

      10, 15, 20, 25, 30, 35, 40, 45,

      50, 55, 60, 65, 70, 75, 80, 85,

      90, 95, 100, 105, 110, 115, 120

    ],

3.点击事件规范：点击事件函数命名方式为 on + 事件名 或者业务名。

onLike: function(event){

}

## 4.4 组件规范：

1. 组件名命名规范：组件在使用时就是小驼峰命名法和下划线命名法结合的组件名，若组件名称为多个单词名拼接而成，采用“\_”连接。组件标签在page页面使用时推荐使用单闭合标签（此条约束对于包含有slot的组件无效）

例如：

/\*=== 计时模式切换的部分 ===\*/

.modeSwitch\_ui{

  width: 100%;

  height: 60%;

  /\* background-color: rgb(235, 255, 52); \*/

  /\*flex布局\*/

  display: flex;

  flex-flow: column nowrap;

  justify-content: center;

  align-items: center;

}

1. 触发事件规范：组件点击触发事件建议用冒号分隔开
2. 组件样式规范：组件在使用时就是小驼峰命名法和下划线命名法结合的组件名，若组件名称为多个单词名拼接而成，采用“\_”连接。

4.标点规范：JS语句无需以分号结束，统一省略分号；JS中一致使用反引号``或单引号' ，不使用双引号；WXML、CSS、JSON中均应使用双引号；CSS属性中冒号中后面用一个空格分隔开；执行一致的缩进（4个空格），执行一致的换行样式(‘unix’)。

# 注释规范：

## WXSS注释规范

每个模块的代码开始前，都要注释好这是写的什么模块什么部分，这个地方的注释采用这种模式的注释/\*=== ===\*/，在这个组（模块）里面，如需注释，则采用/\* \*/的方式。例子如下：

/\*=== 开始按钮部分 ===\*/

.startButton\_ui{

  width: 100%;

  height: 15%;

  /\* background-color: rgb(52, 255, 221); \*/

  /\*flex布局\*/

  display: flex;

  flex-flow: row nowrap;

  align-items: center;

  justify-content: center;

}

## WXML注释规范

## 代码中基本上每个模块都要注释一下，以方便浏览者了解这部分代码是干什么用的，WXML里面的注释模式是<!-- -->。

<!-- 打开侧滑菜单栏的按钮（用view仿制的button） -->

  <view class="btn\_showSideMenu" bindtap="showSideMenu" wx:if="{{isTiming==0}}">

    <image src="/image/菜单.png"></image>

    <view>菜单</view>

  </view>

# 代码走查：

评审对象：设置模块 评审日期：2020/12/29

|  |  |  |
| --- | --- | --- |
| **问题** | **是** | **否，指出问题所在或解释理由** |
| **总体** |  |  |
| 代码编制是否遵照编码规范？ |  | 否，部分一致 |
| 缺陷修改是否完全完成？ |  | 否，部分修改 |
| 所有的代码是否风格保持一致？ | 是 |  |
| **注释** |  |  |
| 所有的注释是否是最新的？ | 是 |  |
| 所有的注释是清楚和正确？ |  | 否，目前未达到完全清楚 |
| 若代码修改注释是否很方便修改？ | 是 |  |
| 所有代码异常处理是否都有注释？ |  | 否，该代码无异常处理 |
| 每一功能目的是否都有注释？ | 是 |  |
| 是否按注释类型格式编写注释？ | 是 |  |
| 代码注释量是否达到了规定值？ | 是 |  |
| **源代码质量** |  |  |
| 所有变量的命名是否依照规则？ | 是 |  |
| 循环嵌套是否优化到最少？ |  | 无嵌套循环 |
| 所有代码是否易懂？ | 是 |  |
| 所有设计要求是否都实现？ |  | 否，部分要求太困难未实现 |
| **其它(根据情况添加)** |  |  |
|  |  |  |
|  |  |  |

开发组长：谢子文 检查人：黄馨

评审对象：我的模块 评审日期：2020/12/29

|  |  |  |
| --- | --- | --- |
| **问题** | **是** | **否，指出问题所在或解释理由** |
| **总体** |  |  |
| 代码编制是否遵照编码规范？ |  | 否，部分一致 |
| 缺陷修改是否完全完成？ |  | 否，部分修改 |
| 所有的代码是否风格保持一致？ | 是 |  |
| **注释** |  |  |
| 所有的注释是否是最新的？ | 是 |  |
| 所有的注释是清楚和正确？ |  |  |
| 若代码修改注释是否很方便修改？ | 是 |  |
| 所有代码异常处理是否都有注释？ |  | 否，无异常处理 |
| 每一功能目的是否都有注释？ | 是 |  |
| 是否按注释类型格式编写注释？ | 是 |  |
| 代码注释量是否达到了规定值？ | 是 |  |
| **源代码质量** |  |  |
| 所有变量的命名是否依照规则？ | 是 |  |
| 循环嵌套是否优化到最少？ |  | 否，无循环嵌套 |
| 所有代码是否易懂？ | 是 |  |
| 所有设计要求是否都实现？ | 是 | 否，部分要求太困难未实现 |
| **其它(根据情况添加)** |  |  |
|  |  |  |
|  |  |  |

开发组长：谢子文 检查人：黄馨

评审对象：专注计时和其他相关模块 评审日期：2020/12/29

|  |  |  |
| --- | --- | --- |
| **问题** | **是** | **否，指出问题所在或解释理由** |
| **总体** |  |  |
| 代码编制是否遵照编码规范？ |  | 否，部分一致 |
| 缺陷修改是否完全完成？ |  | 否，部分修改 |
| 所有的代码是否风格保持一致？ | 是 |  |
| **注释** |  |  |
| 所有的注释是否是最新的？ | 是 |  |
| 所有的注释是清楚和正确？ | 是 |  |
| 若代码修改注释是否很方便修改？ | 是 |  |
| 所有代码异常处理是否都有注释？ |  | 否，因为没有异常处理 |
| 每一功能目的是否都有注释？ | 是 |  |
| 是否按注释类型格式编写注释？ | 是 |  |
| 代码注释量是否达到了规定值？ | 是 |  |
| **源代码质量** |  |  |
| 所有变量的命名是否依照规则？ | 是 |  |
| 循环嵌套是否优化到最少？ |  | 否，无循环嵌套 |
| 所有代码是否易懂？ | 是 |  |
| 所有设计要求是否都实现？ |  | 否，部分要求太困难未实现 |
| **其它(根据情况添加)** |  |  |
|  |  |  |
|  |  |  |

开发组长：黄馨 检查人：梁泽生