G03小组可行性分析报告



Javascript **代码规范**

项目名称： 睡眠小屋

组 号： G 03

组 长： 盛泽文 31701421

组 员： 韩宇 31701413

组 员： 王烨涵 31701423

所在学院： 计算机与计算科学学院

版本记录

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 版本修订日期 | 版本修改人 | 是否通过审核 |
| 1.0 | 2019.5.25 | 盛泽文，王烨涵，韩宇 | 是 |
|  |  |  |  |
|  |  |  |  |

# 规范目的：

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。

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。

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。

# 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：

### 规则

英文单词命名。禁止使用拼音或无意义的字母命名。

### 规则

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

变量及名称开头必须是以下字符中的其中一个字母：a-z或者A-Z中其中一个，例如：va1:function(){}

下划线：\_，例如：var\_a1 \_a1:function(){}

美元符号：$，例如：var$a1  \_a1:function(){}。

### 规则

根据变量（函数）的功能与用途对其进行命名:

Page({  
onload:function(){  
var greet=”hello”;//打招呼

}，

Add:function(firstNum,secondNum){//加法运算函数

Return console.log(firstNum+secondNum);

}  
})

### 规则

变量名（函数名）开头使用小写字母

使用驼峰命名法命名（适用于变量名中包含多个单词）

Page({

Onload:function(){

Var myBlogAddress “http://blog.csdn.net/futruejet”:

### 规则

全局变量使用全大写命名（优点：提高代码可读性）:

Var MONTH\_IN\_1\_YEAR=12;//一年有12个月

Page（{

Onload:function(){

}  
})

## 特殊规则：

### 【规则2-1】

### 规则

没有使用var关键字声明变量（注：调试程序时会报错）

Page({

Onload:function(){

greet=”Hello”;(出错)

}

})

### 【规则2-2】

### 规则

变量名（函数名）中出现空格（报错）

Page（{

onload:function(){  
var gre et=”hello”;(出错）

}

})

### 【规则2-3】

### 规则

变量名首个字符为数字（报错）

Page（{

onload:function(){  
var 00=10;(出错）

Var 0a=10;(出错）

}

})

### 【规则2-4】

### 规则

### 函数名首个字符为数字或直接用数字命名

Page（{

onload:function(){  
this.0();(出错）

}

0:function(){

Console.log(“0”)

}

})

### 【规则2-5】

### 规则

变量名称随意，降低代码可读性

Page（{

onload:function(){  
var a1=1;

}

})

# 编码规范：

## 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.控制每行HTML的代码数量在50个字符以内，方便阅读浏览，多余的代码进行换行处理，标签所带属性每个属性间进行换行。

<v-music

wx:if="{{classic.type===200}}"

img="{{classic.img}}"

content="{{classic.content}}"

</v-music>

1. 合理展现分离内容，不要使用内联样式。

推荐使用：<image class=”tag”</image>

## 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，字体大小和边框等使用px，开发者根据实际情况而定。

例如：

width: 100rpx;

font-size: 14px;

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。

.v-tag{

width: 100%;

}

.v-container{

width: 100%;

}

}

1. 尽量使用简写属性，并且同一属性放置在一起，避免散乱

/\*\*使用简写属性\*\*/

.v-image{

margin: 0 auto;

}

/\*\*同一属性放在一块\*\*/

.v-tag{

margin-left: 10rpx;

margin-right: 10rpx

}。

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

.container{

disaplay: flex;

flex-dirextion: row

}

## JS规范

1.命名规范：变量名以及函数名统一采用驼峰命名法，正常情况下函数名前缀需加上清晰的动词表示函数功能，私有函数或者属性以下划线开头表明。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

let imageContent = res.data

//函数命名

getInfo:function(){

return '';

} //私有函数

\_getInfo:function(){

1. 回调函数规范：回调函数统一使用Promise函数的方式进行编写，回调成功的参数统一为res，错误参数为err。

// promise 处理回调

let back = new Promise((resolve, reject) => {

if (/\* 异步操作成功 \*/){

resolve(value);

} else {

reject(error);

}});

back.then((res) => {

console.log('成功回调！', res);

}).catch((err) => {

console.log('失败回调！', error);

});

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){

}

## 组件规范：

1. 组件名命名规范：组件在使用时命名以 “v-”为开头的组件名，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）
2. 触发事件规范：组件点击触发事件建议用冒号分隔开
3. 组件样式规范：所有组件样式均应采用类的写法，且命名必须以 v- 开头，不允许使用内联样式以及id样式

.v-container{

disaplay: flex;

flex-dirextion: row

}

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

成组的wxss规则之间用块状注释。请勿在代码后面直接注释。

/\*\* 修改button默认的点击态样式类\*\*/

.button-hover {

background-color: red;

}

## wxml注释规范

## 除组件外的其他块级元素，均需注释出其功能，并在其上下空出一行与其他代码进行区分。

<view>...</view>

//导航栏

<view>...</view>

<view>...</view

# 代码走查：

目前只挑选了Index，Logs，List三份代码进行走查。

评审对象：Index 评审日期：

|  |  |  |
| --- | --- | --- |
| **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？ | 是 |  |
| 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：盛泽文 检查人：韩宇

评审对象：Logs 评审日期：

|  |  |  |
| --- | --- | --- |
| **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？ | 是 |  |
| 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：盛泽文 检查人：盛泽文

评审对象：List 评审日期：

|  |  |  |
| --- | --- | --- |
| **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？ | 是 |  |
| 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：盛泽文 检查人：王烨涵