**手机安卓端app代码规范**

****

|  |  |  |
| --- | --- | --- |
| 文件状态：  [ ] 草稿  [ ] 正式发布  [√] 正在修改 | 文件标识： | SE2018春-G08-迷城逃亡app代码规范 |
| 当前版本： | 0.1.1 |
| 作者： | 吴子乔，陈栩,石梦韬 |
| 完成日期： | 2018-06-02 |

# 版本历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本/状态 | 参与者 | 起止日期 | 备注 | 审核人 |
| 0.1.1 | 吴子乔 | 2018-06-01至  2018-06-02 | 代码规范进行编写 | 陈栩,石梦韬 |
|  |  |  |  |  |

目录

[版本历史 2](#_Toc515807614)

[一、 文件组织规则 3](#_Toc515807615)

[1.1命名 3](#_Toc515807616)

[1.2目录 3](#_Toc515807617)

[二、 代码组织规则 3](#_Toc515807618)

[2.1空行 3](#_Toc515807619)

[2.2换行 3](#_Toc515807620)

[2.3空格 4](#_Toc515807621)

[2.4对齐 4](#_Toc515807622)

[2.5就近原则 4](#_Toc515807623)

[三、 命名规则 4](#_Toc515807624)

[3.1自注释 4](#_Toc515807625)

[3.2变量 4](#_Toc515807626)

[3.3函数 5](#_Toc515807627)

[3.4类型 5](#_Toc515807628)

[四、 注释规则 5](#_Toc515807629)

[4.1有效性 5](#_Toc515807630)

[4.2普通注释 5](#_Toc515807631)

[4.3定义 5](#_Toc515807632)

[4.4模块 5](#_Toc515807633)

[五、 维护规则 6](#_Toc515807634)

[5.1消除警告 6](#_Toc515807635)

[5.2代码修改 6](#_Toc515807636)

# 文件组织规则

## 1.1命名

所有的目录和文件名使用单词组合，在单词缺乏的情况下引用汉语拼音组合。

如 GameQuit //离开游戏代码操作

双驼峰类型

## 1.2目录

对于目录名采用英文单词解释组合。

如 c# //c#操作脚本

# 代码组织规则

## 2.1空行

在类、结构、联合、函数、枚举等定义结构后，应该加空行。

代码布局习惯应该符合c#编程代码风格。

如

void Start()

{

//Ensure that there is only one gameQuit in the Scene，即使加载了下个场景Scene

GameObject[] gameQuits = GameObject.FindGameObjectsWithTag("GameQuit");

if (gameQuits.Length == 2)

{

Destroy(this.gameObject);

}

DontDestroyOnLoad(this.gameObject);

}

## 2.2换行

每行代码只写一条语句。

拆分复杂的复合表达式。

如

void Start()

{

//Ensure that there is only one gameQuit in the Scene，即使加载了下个场景Scene

GameObject[] gameQuits = GameObject.FindGameObjectsWithTag("GameQuit");

if (gameQuits.Length == 2)

{

Destroy(this.gameObject);

}

DontDestroyOnLoad(this.gameObject);

}

## 2.3空格

对于空格的留取与操作应符合c#代码习惯

如 gameQuits.Length == 2 //在 == 号前后都留取空格

## 2.4对齐

对于代码行的对齐操作，应该以 visual studio内的自动对齐与格式矫正为准。

## 2.5就近原则

较为紧密的代码应该尽可能相邻

变量应在其定义的同时初始化

如 public static int Gamecrl = 0;

# 命名规则

整体采用C#代码风格，提高效率

## 3.1自注释

对于所有的变量使用都应该注释其作用，对于所有方法的定义都应注释其含义及作用，同时在调用方法时都应该注释调用原因

如 public int tag //标志位使用

## 3.2变量

在定义全局静态变量时，采用大写字母开头的形式以单词组合命名。

在定义其他变量时，采用全小写形式单词组合命名。

如 public static int Gamecrl = 0; //定义游戏控制变量

## 3.3函数

使用“动词”或者“动词+名词”的形式命名函数名。

如 void Start()

## 3.4类型

在定义类型时，应该以c#变量类型命名规则相关。

如 int 、float 等

# 注释规则

## 4.1有效性

注释内容要清楚、明了，含义准确，防止注释的二义性。

在代码的功能、意图层次上进行注释，提供有用、额外的信息。

对于已经有充分自注释的代码无需再注释。

## 4.2普通注释

在函数内部使用C#注释风格，对于方法的注释写在方法内第一行，对于变量及对象的注释写在其上方或者下方。

## 4.3定义

对于类、方法、结构、枚举、自定义类型、变量等定义需要加上注释，解释其用途。

## 4.4模块

对于有相同用途或者定义的将其分入同一模块注释。

# 维护规则

## 5.1消除警告

使用和项目组内统一的编译器版本和选项。

重视编译器的警告，设法尽量消除警告。

## 5.2代码修改

维护时不要修改已经定义好的函数，而是应该定义新的函数。

禁止随意更改其他模块或者系统的相关设置和配置。

去掉函数或者程序中无用的代码，防止垃圾代码。