# 编码规范说明文档

2014-05-14

# 一、命名规范

## 1.1.1类命名

首字母大写，之后每个单词首字母都大写。

使用能够反映类功能的名词短语。

文件的名称应与类名一致。

## 1.1.2特殊类命名

视图控制器的子类应添加后缀 ”ViewController” 或者 ”Controller”。

视图的子类应添加后缀 “View”。

按钮的子类应添加后缀 “Button”。

……

## 1.1.3分类（类别）命名

与类命名相同，此外需添加要扩展的类名和 “+”。

## 1.1.4协议（委托）命名

与类命名相同，此外需添加 ”Delegate” 后缀。

## 1.1.5方法命名

首字母小写，之后每个单词首字母都大写。

使用能够反应方法功能的动词短语。

## 1.1.6方法、函数参数命名

首字母小写，之后每个单词首字母都大写。

具有足够的说明性。

## 1.1.7变量命名

首字母小写，之后每个单词首字母都大写。

具有足够的说明性。

成员变量添加 ”\_” 前缀。

不得使用匈牙利命名法。

## 1.1.8常量命名

常量（预定义，局部常量等）使用小写 ”k” 开头的驼峰法。

枚举类型命名首字母大写，之后每个单词首字母都大写，最后加”s”。

枚举变量使用枚举类型去掉 ”s” 作为前缀，每个单词首字母大写，中间不允许加下划线。

## 1.1.9图片命名

使用英文，首字母大写，之后每个单词首字母都大写。

添加模块名作为前缀，避免冲突。

图片资源应该与类文件一样，按模块分组放置。

## 1.2.0分组命名

使用英文，首字母大写，之后每个单词首字母都大写。

每个分组使用模块的名字。

引用第三方代码库统一放在 ”Library” 分组下。

使用的公共组件统一放在 ”Common” 分组下。

视图控制器及AppDelegate统一放在 ”Controllers” 分组下。

# 二、书写规范

## 2.1.1注释

多行注释尽量使用/\*……\*/。

单行注释使用// ，且//后应加空格与描述相隔开。

类文件都包含类文件头，要说明文件名、作者、创建时间、类文件说明等。

每个头文件中的Public方法都应该提供方法说明，参数，返回值等注释。

当使用if、for、switch语句时，需要用注释说明用意。

如果类的实例可以被多个线程访问，记得注释多线程条件下的使用规则。

每个接口、类别以及协议应该注释,以描述它的目的及作用。

使用 | 来引用注释中的变量名及符号名而不是使用引号。

## 2.1.2方法

留一个空格在-或+和返回类型之间，但参数列表里的参数之间不要留间隔，每个冒号之前都必须有对该参数的说明词。

如果参数过多，推荐每个参数各占一行。使用多行的情况下，在参数前加冒号用于对齐。

## 2.1.3其他规范

操作符前后都要加空格。

避免相同的代码段在多个地方出现。

语句嵌套层次不得超过3层或3层以上。

每个实现文件建议在500行以内，不能超过1000行，超过之后应考虑通过抽象。类对代码进行重构。

及时删除或注释掉无用的代码，如：NSLog语句。

方法调用时，所有参数应该在同一行，或者每行一个参数,以冒号对齐。

应尽量使用“代理设计模式”使类与类之间解耦。

开发应尽量使用MVC设计原则。

# 三、技术说明

## 3.1.1文件的扩展名

.h, C/C++/Objective-C的头文件。

.m, Ojbective-C实现文件。

.mm, Ojbective-C++的实现文件。

.cc, 纯C++的实现文件。

.c, 纯C的实现文件。

## 3.1.2文件引用

#import Ojbective-C/Objective-C++头文件。

#include C/C++头文件。