代码规范

1. 引用第三方库规范

**1、引用第三方框架，必须编写一个readme文档附带于第三方框架的文件夹下，主要编写植入该框架的作用与植入步骤。**

比如：

AFNetworking

框架的作用：用于请求接口，获取网络数据。

步骤：直接将AFNetworking文件夹添加到Lib文件夹下即可。

**2、建议大家使用CocoaPods（除非CocoaPods上没有此项目）**

好处：

1、解决了项目之间依赖

2、第三方库更新方便

因为直接导入源码的话，需要修改项目配置，且有时候不同框架要求不同的项目配置，这个比较麻烦。 并且需要导入很多系统的动态库。

1. 工程规范

**1、为了避免文件杂乱，物理文件应该保持和 Xcode 项目文件同步，Xcode 创建的任何组（group）都必须在文件系统有相应的映射。**

比如：

qingsongchou/

    Home                 代码

        /Controller             控制器

        /View                   视图

        /Model                  数据模型

        /Tool                   工具类

        /Catergory              扩展类

        /Database               数据库

    Lib                         第三方库

    Frameworks                  系统库

1. 命名规范

**1、项目包名统一格式为：com.qschou.social。**

**2、为了避免和苹果SDK或其他第三方框架冲突，命名类的时候，需要统一加命名空间前缀，前缀以项目名每个单词的第一个字母拼接而成。**

比如：QSCUser

**3、所有变量、方法名、协议、常量、宏等都需要命名清晰、可读性高，主要使用驼峰命名法结合英文单词全拼法。**

（1）变量：

NSDictionary \*userDictionary;//规范

NSDictionary \*userDict;//不规范

**4、整个工程的命名风格要保持一致性，每个人的代码风格保持高度统一。**

比如：不能在ClassA中使用count而在ClassB中使用totalCount来返回集合的个数。

**5、类名以大写字母开头，方法名、变量以小写字母开头，常量、宏以全大写字母命名。**

//类名：

QSCUser \*user;//规范

qscUser \*user;//不规范

//方法名：

- (void)setUserName;//规范

- (void)SetUserName;//不规范

//变量：

NSString \*userName;//规范

NSString \*UserName;//不规范

//常量：

NSInteger kUSERSEX;//规范

NSInteger UserSex;//不规范

//宏：

NSInteger QSCUSERSEX 0;//规范

NSInteger UserSex 0;//不规范

**6、如果方法是为了获取对象的一个属性值，直接用属性名称来命名这个方法，注意不要添加get或者其他的动词前缀。**

- (NSSize)cellSize;//规范

- (NSSize)getCellSize;//不规范

**7、对于有多个参数的方法，务必在每一个参数前都添加关键词，关键词应当清晰说明参数的作用。**

- (void)insertObject:(id)object atIndex:(NSInteger)index;//规范

- (void)insert:(id)object at:(NSInteger)index;//不规范

**8、委托方法的第一个参数必须是触发它的对象，除非委托方法只有一个参数。**

//两个参数

- (BOOL)application:(NSApplication \*)application openFile:(NSString \*)filename;

//一个参数

- (BOOL)windowShouldClose:(UIWindow \*)window;

**9、根据委托方法触发的时机和目的，多使用should、will、did等关键词。**

比如：

- (void)scrollViewDidScroll:(UIScrollView \*)scrollView;

**10、类对象的属性值如果是整型常量，尽量使用枚举。**

比如：

//微爱类型

typedef NS\_ENUM(NSInteger, QSCLoveType){

QSCLoveTypeSickness = 3349, // 大病

QSCLoveTypeHelp, // 灾难救助

QSCLoveTypeAnimal, // 动物保护

QSCLoveTypeStudent, //扶贫助学

QSCLoveTypeOther = 3359 //其他

};

**11、通知要尽可能地表示出发生的事件，通知的命名范式为：[触发通知的类名] + [Did|Will|should] + [动作] + Notification**

比如：NSKeyboardWillShowNotification

**12、初始化对象的方法，分为实例方法与类方法。**

（1）实例方法范式为：initWith+[类名|属性名] + [类对象|属性值]

（2）类方法范式为：[类名小写] + With + [属性名] + [属性值]

比如：

//用户信息实例方法

- (instancetype)initWithUserName:(NSString \*)userName userSex:(int)userSex;

- //用户信息类方法

+ (instancetype)userWithUserName:(NSString \*)userName userSex:(int)userSex;

**13、Resource素材资源命名方法。**

范式：［模块名］＋［下划线］＋［类型］＋［下划线］＋［功能］

比如：

home\_icon\_map //首页地图图标

me\_icon\_edit //编辑用户信息图标

common\_icon\_save //公用的保存图标

//类型列表

(1) btn //按钮

(2) hlbtn //高亮按钮

(3) bg //背景图片

(4) hlbg //高亮背景图片

(5) img //图片

(6) icon //图标

**14、所有视图控件、视图组件的类名称必须以后缀结尾。**

后缀类型列表如下：

（1）UIViewController后缀为“ViewController”；

（2）UIView后缀为“View”；

（3）UIButton后缀为“Button”；

（4）UILabel后缀为“Label”。

**15、项目文件名、文件夹名、图片名、变量名等不要使用汉字命名，尽量使用英文命名**

1. 注释规范

**总体规则：尽量让代码可以自表述，而不是依赖注释。**

**1、合理的注释。好的注释不仅能让人轻松读懂你的程序，还能提升代码的规格。注释是为了让别人看懂代码，也是让你自己看懂代码。**

**2、注释内容条例。**

（1）特殊的property均需要注释；

（2）所有自定义的方法需要注释；

（3）复杂的代码片段需要加以注释。

**3、类或文件的良好注释通常包涵：类文件名、项目名、作者信息、创建日期、版权信息、类文件包含的内容及作用。**

比如：

//

// QSCAccount.h

// qingsongchou

//

// Chai on 15/9/15.

// Copyright (c) 2015年 Chai. All rights reserved.

// 用户对象

**4、方法或函数的良好注释通常包涵：代码的作用、参数、返回值。（可使用VVDocumenter插件进行注释）**

比如：

    /\*\*

     \* 获取城市列表数据

     \* @param 无

     \* @return 城市列表数据cityArray

     \*/

五．排版规范

**1、类property属性排版规范**

规则：[property] + [空格] + [类型] + [空格] + [属性名]

比如：

**@property**(nonactomic, strong) NSNumber **\***score;*//分数*

**@property**(nonactomic, copy) NSString **\***name;*//名称*

**@property**(nonactomic, assign) BOOL isUpdate;*//是否正确*

**2、类实现文件排版规范**

规则：方法类型与返回类型之间要有空格。

比如：

//错误

-(void)methodName:(NSString \*)string;

//正确

- (void)methodName:(NSString \*)string;

**3、方法排版规范二**

规则：参数在两个以内，不需要分行；参数在三个及三个以上，每个参数都需要分行，并且以“:”对齐。

比如：

//两个参数

- (void)initWithUserName:(NSString \*)userName userSex:(int)userSex;

//三个参数（参数间以“:”对齐）

- (void)initWithUserName:(NSString \*)userName

userSex:(int)userSex

userAge:(int)userAge;

**4、方法排版规范三**

规则：两个方法间，空一行。

比如：

//初始化

- (void)initWithUserName:(NSString \*)userName;

//初始化

- (void)initWithUserSex:(int)userSex;

**5、import排版规范**

规则：import文件时，以块来划分文件类型。

比如：

//Views

#import "QSCProjectView.h"

#import "QSCBrandCell.h"

//Models

#import "QSCProject.h"

#import "QSCBrand.h"

//Controllers

#import "QSCRedCrossController.h"

#import "QSCLoginController.h"

**6、类实现文件排版规范**

规则：类实现文件以块来划分代码，范式为：[#pragma mark] + [-] + [块名]。块分为：

（1）生命周期LifeCycle

（2）视图View

（3）事件处理Event

（4）网络服务Webservice

（5）协议回调Delegate

（6）页面跳转Push

比如：

#pragma mark -LifeCycle

//视图加载完毕

- (void)viewDidLoad {}

#pragma mark -Event

//点击返回按钮

- (void)backButtonClick{}

#pragma mark -load

//从服务器获取数据

- (void)loadData

#pragma mark -Delegate

//直播点击回调

- (void)LiveMainModalViewTapBtnWithType {}

#pragma mark -Push

//跳转到用户信息界面

- (void)pushProductInfo {}