IOS开发代码规范

**命名**

命名规则对于维护代码来说是非常重要的,。Objective-C方法名往往很长，不过这也有好处，让很多注释变得毫无意义。

驼峰法是Objective-C社区的标准。

驼峰法分小驼峰法和大驼峰法。小驼峰法：除第一个单词之外，其他单词首字母大写。大驼峰法相比小驼峰法，大驼峰法把第一个单词的首字母也大写了。

**1.基本原则**

**1.1 清晰**

又清晰又简洁是最好的了，但简洁不如清晰重要。总的讲不要使用单词的简写，除了非常常用的简写以外，尽量使用单词全称。API的名称不要有歧义，一看你的API就知道是以什么方式做了什么事情，不要让人有疑问

**1.2 一致性**

做某个事情代码通常都叫这个名字，比如tag、setStringValue，那么你也这么叫。你在不确定怎么起名字的时候，可以参考一些常用的名字：[Method Arguments](https://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/CodingGuidelines/Articles/NamingMethods.html" \l "//apple_ref/doc/uid/20001282-1001865" \t "_blank)

**2. 类命名**

类名应该用大写开头的大驼峰命名法。类名中应该包含一个或多个名词来说明这个类（或者类的对象）是做什么的。

在应用级别的代码里，尽量不要使用带前缀的类名。每个类都有相同的前缀不能提高可读性。不过如果是编写多个应用间的共享代码，前缀就是可接受并推荐的做法了(型如 JKPhotoBrowser )。

**文件组织结构**

1. 类文件组织

iOS工程文件结构分物理结构和逻辑结构，建议逻辑结构和物理结构保持一致，以便方便有效地管理类文件。类文件组织要遵循以下两大原则：

基于MVC设计模式原则，至少要保证controller与数据处理，网络请求相对独立

基于功能模块原则，功能模块分包括数据/网络处理，UI前端界面两部分，数据/网络处理应该在数据/网络处理的框架下，而UI前端界面比如用户中心，消息中心，它们的专有的controller，view等应该在属于文件夹。还会遇到一些公共的view，可以开辟出公共的文件夹来管理

在实际中使用中，项目实际负责人可以结合项目特点灵活使用，但基本的原则一定要保持，保持良好的类文件组织结构，对团队有益无害。

2. 图片资源文件组织

图片资源文件，强烈建议采用Images.xcassets管理，尽量少用自己创建的文件夹管理。

使用Images.xcassets的优势很多，具体可以查阅读相关文献资料，这里只从工程管理上说一点，在Images.xcassets中添加图片资源，不会对project文件造成改变，而直接在文件夹里添加图片文件，每次都会对project文件造成改变，因此使用Images.xcassets管理图片资源可以减少project冲突的次数。

iOS代码规范  
1 删除多余的空行  
\* 所有方法与方法之间空1行  
\* 所有代码块之间空1行  
2 删除多余的注释  
\* 删除注释掉的代码  
\* 删除没有意义的注释  
3 删除多余的方法  
\* 如果方法没有使用到，请删除它  
\* 如果方法没有执行任何业务逻辑，请删除它或者给出一定注释  
4 删除未被使用的资源文件  
5 添加必要的注释  
\* 所有 .h 文件中的property 需要给出注释  
\* 所有自定义的方法需要给出注释  
\* 比较大的代码块需要给出注释  
\* 所有代码中出现的阿拉伯数字需要给出注释  
\* 程序中出现加密／解密 逻辑的操作地方，需要给出注释说明过程（无论是系统还是自定义）  
6 整体代码风格需要统一  
\* 代码后面的”{“ 不需要单独占用一行  
\* 逻辑运算符 与 代码之前空一格  
\* “#pragma mark -” 与下面的代码之前不要空行  
\* 遵循一般性的代码规范

iOS通用规则  
1 、下面所有规则对第三方类库无约束  
\* 所有类、方法、属性等命名，做到见名知意，采用驼峰式命名规则  
\* 根据资源类型或者所属业务逻辑对项目资源进行分组，使得整个项目结构清晰明了  
\* 整个项目保持一种代码书写风格（这个风格由无锡团队根据自己编码习惯来定），让你的代码变的优雅！  
2. 命名规范  
\* 所有类名称以项目工程开头命名，eg：“XP”、“ZJG”、“SZ”  
\* 针对不同视图控制器，在末尾添加后缀，eg：  
\* UIViewController  后缀添加“ViewController”  
\* UIView 后缀添加“View”  
\* UIButton 后缀添加“Button"  
\* UILabel 后缀添加“Label"  
3. 单页代码最好控制在800行以内，每个方法最好不要超过100行，过多建议对代码进行重构  
4. 相同的逻辑方法定义避免在多个地方出现，尽量将公用的类、方法抽取出来  
5. 删除未被使用的代码，不要大片注释未被使用的代码，确定代码不会使用，请及时删除  
6. 对其他项目中copy过来的代码，根据具体需要更新代码风格，及时删除未被使用的代码  
7. 项目中所有Group或者文件名称（图片名字等），不要使用汉字命名，尽量使用英文命名，国内特有名词可以使用拼音。  
8. 项目中所有Group都需要在项目目录中存在一个真实的目录，Group中的文件与真实目录中文件一一对应。  
9. 请在项目中写必要代码的注释  
10. 请多使用 #pragma mark - Mark Name 对方法进行分组 eg:  
\* #pragma mark - View lifeCycle  
\* #pragma mark - View lifeTerm  
\* #pragma mark - Init methods  
\* #pragma mark - Action methods  
\* #pragma mark - Common methods  
\* #pragma mark - UIActionSheetDelegate  
\* #pragma mark - UIImagePickerControllerDelegate  
\* #pragma mark - UITableViewDelegate Methods  
\* #pragma mark - UITableViewDataSource Methods  
\* #pragma mark - UIScrollViewDelegate Methods  
\* #pragma mark - UITextFieldDelegate Methods  
\* #pragma mark - UITextViewDelegate Methods