# iOS开发规范

iOS开发规范 1

简介 1

目录结构 1

命名规范 2

整体命名说明 2

ViewController（视图控制器） 3

Components（组件/自定义控件） 3

Resources（资源文件） 3

Cell（可重用元素） 3

Entity（实体类） 3

Config.h CustomUtil.h文件 4

注释规范 4

文档生成 4

开源工具使用要求 4

xcode插件使用 4

常用工具使用 5

## 简介

为了使开发无障碍化，同时加速应用的各个版本开发，订立以下规范，同时说明开发目录结构，以便开发者能够快速了解本框架。建议本公司IOS开发者采用该框架，尽量在开发规范下进行开发。

## 目录结构

WorkPlace:

--Base //项目主项目

----Base //核心框架主目录

------Resource //常用的资源目录，包括图片、声音、xml文档、样式和数据

------Components //常用组件（自定义控件）以及所有控件核心基类

------Config //常用的宏命令以及语言部分。。

------Entity //实体类基类

------Network //与webservice交互的、本地数据库的以及业务基类

------Util //常用的工具类（包含神器、网络图片、xml、soap、form、崩溃报告）

------Supporting Files //配置文件以及，appDelegate基类，用于加载基础数据

----APP //项目主目录

------Components //组件目录，所有的组件（自定义控件）均放在此处

------Resource //资源目录，包括图片、声音、xml文档、样式和数据

------Cell //各种可重用的元素目录，一般包括UITableViewCell子类和UICollectionViewCell子类

------Entity //实体类目录，包括Coredata产生的实体和自定义的实体类

------ShareHandle //单例目录

------Viewcontroller //视图控制器目录

------CustomUtil //配置文件，一般存放用户信息等本地化数据

------Service //业务请求文件，所有的业务请求，包括本地数据库、Webservice等

------Config //配置文件

------AppDelegate //应用程序启动文件

------Basexcdatamodeld //数据库模型文件

----BaseTests //测试主目录

----Framework //IOS的类库框架

----Prouducts //编译出来的archive

--Pods //pods，所有的开源代码包管理工具，便于开发者快速导入开源插件

## 命名规范

### 整体命名说明

1、采用驼峰命名规则，单页代码长度尽可能少于200行，如有特殊情况无法避免的除外，所有类的定义必须继承核心基类，凡是不够用的，需审核后在核心库中添加基类

2、变量定义，

①尽量减少全局变量的定义，严禁使用跨类的全局变量，使用单例代替，全局变量命名定义必须以下划线为开头（属性除外），第一个字母小写，下划线+类型名+业务名

②局部变量定义：业务名+类型名

3、定义常量，

①任何常量禁止使用，要用宏来代替

②命名规则：U\_+业务类型 +业务名

③包含范围：enum、常量

4、方法（函数）定义

①采用驼峰法，第一个字母小写

②类方法定义和成员方法定义规则可自行拟定，但是要附带词义，严禁无关的词义出现，当词义发生改变，必须修改方法名

③将不必要暴露的函数声明写在实现文件中，外部暴露的函数写在头文件里

④方法中代码行数不应该超过50行，多则分散到小的函数块儿中

5、类名定义

①命名规则：模块名+业务名+类型组成

②第一个字母必须大写，所使用的自定义类应该全部继承基类统一管理

6、协议的定义

①命名规则：I+业务名+Delegate/Protocal

②尽量将所有的方法定义为可选功能，如果非要必选，则选择必选

8、category的定义

①命名规则：小括号里写业务即可，第一个字母小写

9、block的定义

①允许使用block，第一个字母小写

### ViewController（视图控制器）

1、所有有关业务处理、视图链接以及页面跳转逻辑的均放在APP->ViewController，严禁放在其他目录

2、允许分模块将模块目录放进ViewController目录

### Components（组件/自定义控件）

1、所有的只跟数据打交道的视图，定义在此，严禁定义在ViewController中

2、定义接口，仅有部分泄露，凡是有反馈的，均需要通过协议和委托来实现

### Resources（资源文件）

1、严禁将图片等多媒体资源放在其他目录，同时如果有需要共享的，需要经过审批进入核心目录，以完成框架的升级

2、资源文件中图片资源需要设置两份@2x作为后缀

### Cell（可重用元素）

1、所有的可重用元素均放在Cell目录，好找，所有类必须继承核心库中基类

2、需要自定义视图的，单独创建与类名相同的xib文件，在其中使用堆叠技术，累出界面

3、和普通组件一样，仅暴露数据接口，严禁暴露组件接口，凡是需要与外部统筹资源需要交互的，必须使用委托和协议

### Entity（实体类）

1、所有的实体类必须继承核心库的基类

2、允许给Entity核心基类添加多的方法，但是想要入库需要审核

3、命名方法为Entity放在前，后面跟业务逻辑名

### Config.h CustomUtil.h文件

1、所有的配置文件均写于此

2、常量需要写上注释

## 注释规范

代码注释采用xcode插件注释方法

## 文档生成

文档生成使用doxygen

## 开源工具使用要求

1、网络处理：ASIHttpRequest

2、数据库处理：MagicalRecord

3、Socket处理：CocoaAsyncSocket

4、拼音转换：PinYin4Objc

5、加载插件：MBProgressHUD

6、评论提醒：Appirater

7、表单切换：TPKeyboardAvoiding

8、crash采集：KSCrash

9、界面解析：Reveal

10、图片缓存：SDWebImage

## xcode插件使用

1、cocoapods 源码包管理工具

2、KSImageNamed 图片提示

3、VVDocumenter 注释生成

4、OMColorSense 颜色显示

5、FuzzyAutocomplete 代码提示

6、Peckham 头文件补全

7、GitDiff 文件改变比较

## 常用工具使用

1、Cornerstone SVN源码版本管理工具

2、WireShark 网络协议分析

3、Doxygen 文档生成工具

4、Dash API查询工具

5、Reveal 界面结构分析工具

6、PaintCode 绘图生成代码工具

7、Prepo 图片批量处理工具

8、Liya 数据库查看工具

9、VisualJson json快速查看工具

10、echo 网络API测试工具

11、Apns Pusher Apns证书测试工具

12、idaq 动态反编译工具

13、Pxcook快速标注工具