目录

[1.命名规范 2](#_Toc475446587)

[1.1类型名简写： 2](#_Toc475446588)

[1.1.1UIKit类 2](#_Toc475446589)

[1.1.2Foundation类 2](#_Toc475446590)

[2.代码组织 5](#_Toc475446591)

[2.1#pragma mark - 使用 5](#_Toc475446592)

[3.字面值 6](#_Toc475446593)

[4.三目运算符 6](#_Toc475446594)

[5.常量 6](#_Toc475446595)

# 1.命名规范

## 1.1类型名简写：

### 1.1.1UIKit类

UIAlertView alertV

UIAlertController alertContr

UIImageView imageV

UINavigationController navContr

UIPickerView pickerV

UIScrollView scrollV

UITableView tableV

UITextField tf

UITextView tv

UIWebView webV

View不在开头简写成 V

例如：UITableView tableV

Controller不在开头简写成 Contr

例如：UIAlertController alertContr

其余类型名不简写，只把 UI 去掉

例如：UIButton button

### 1.1.2Foundation类

NSAttributedString attriStr

NSDictionary dic

NSString str

NSMutableDictionary mutableDic

NSMutableString mutableStr

Attributed 不在开头简写成 attri

例如：NSAttributedString attriStr

其余类型名不简写，只把 NS 去掉

例如：NSArray array

Bool is

NSInteger int

CGFloat float

变量命名：类型名+作用，首字母小写（登录 login 为例）

例如：

@property (nonatomic, strong) UITextField \*tfLogin;

@property (nonatomic, copy) NSString \*strLogin;<NSString 类必须有 copy>

@property (nonatomic, assign) Bool \*isLoginSuccess;

@property (nonatomic, strong) UITextField \*tfLogin;

<(类型) 空格 (\*)(变量名)>

类命名：作用+父类标识，首字母大写（登录 login 为例）<防止Block在不同类中重名>

例如：

LoginVContr

LoginImageV

定义 Block 命名：block 功能\_所在类类名（ViewController类，登录 login 为例）

typedef void(^LoginBlock\_ViewController)(NSString \*);

定义枚举和枚举内标示位：（ViewController类，登录 login 为例）<防止枚举在不同类中重名>

枚举名：enum功能\_枚举所在类名

枚举内标识位：标识位功能\_枚举所在类

typedef NS\_ENUM(NSInteger, LoginEnum\_ViewController) {

mobileLogin\_ViewController,

usernameLogin\_ViewController,

};

方法命名：尽可能用方法名把方法的功能说清楚

私有方法加前缀 pri\_

-(或+) 空格 (返回类型)(方法名)(参数1) 空格 (参数2)... 空格 {<大括号跟在方法名之后，不要另起一行>

}

例如：

- (NSString \*)getTheBase64String:(NSString \*)string {

}

私有方法

- (NSString \*)pri\_getTheBase64String:(NSString \*)string {

}

# 2.代码组织

## 2.1#pragma mark - 使用

1. #pragma mark - Lifecycle(生命周期)
2. - (instancetype)init {}
3. - (**void**)dealloc {}
4. - (**void**)viewDidLoad {}
5. - (**void**)viewWillAppear:(**BOOL**)animated {}
6. - (**void**)didReceiveMemoryWarning {}
7. #pragma mark - CustomizeMethod(自定义方法)
8. #pragma mark - Public(公开方法)
9. - (**void**)publicMethod {}
10. #pragma mark - Private  (私有方法)
11. - (**void**)pri\_privateMethod {}
12. #pragma mark - lazyLoading(懒加载)
13. - (UIButton \*)buttonLogin {}
14. #pragma mark - Action(各种事件响应的方法)
15. - (**void**)buttonLoginAction {}
16. #pragma mark - Delegate  (代理)
17. #pragma mark - UITextFieldDelegate
18. #pragma mark - UITableViewDataSource
19. #pragma mark - UITableViewDelegate

# 3.字面值

NSString、NSDictionary、NSArray 和 NSNumber 创建时使用 @“”/@{}/@[]/@()

例如：

NSString \*strName = @“Tom”;

NSArray \*arrayNames = @[@"Brian", @"Matt", @"Chris", @"Alex", @"Paul"];

NSDictionary \*dicProductManagers = @{@"iPhone": @"Kat", @"iPad": @"Kam"};

NSNumber \*numberBuildingStreet = @(10018);

# 4.三目运算符

三目运算符尽量避免嵌套，如果必须嵌套要用（）来区分

例如：result = a > b ? (x = (c > d ? c : d)) : y;

# 5.常量

常量是容易重复被使用和无需通过查找和代替就能快速修改值。常量应该使用static来声明而不是使用#define，除非显式地使用宏。

static NSString \*strProductName = “QPM”;