Code Style Guide [iOS]

Clarity at the point of useis your most important goal.

Module Name:

模块定义命名：

avoid use Binance/BNC prefix

- Module 包含界面且实现某部分具体业务功能的模块叫 Module | A module that contains an interface and implements a specific part of the business function is call Module
- Service. 只包含具体业务逻辑的模块叫Service | Modules that contain only specific business logic are called Service
- SDK. 不包含业务逻辑， 提供基础服务的组件叫SDK | Components that do not contain business logic and provide basic services are called SDKs
    - 禁止出现类似Common、 Base、 Foundtaion的SDK | Should avoid naming of SDK like Common/Base/Foundation
    - SDK拆封粒度要细， 禁止完全不相关的功能出现在同一个SDK | SDK unpacking granularity to prohibit completely unrelated features from appearing in the same SDK
- Kit. 包括多个类似基础服务的组件叫Kit | The component that includes several similar basic services is called Kit

Module File directory Recommend:

需要考虑按照功能来分文件

1. Core
    a. ViewControllers
    b. Views
    c. Coordinators
    d. Models
    e. Services
    f. Extensions
    g. Constants
2. Subspec
3. General (Dedicated to put common files)
    a. All URLs
    b. All Analytics Keys
    c. All Enum
    d. All ABTest
    e. .etc

Class/Struct/Enum Name:

Lite/Funds+Deposit+ViewController

Lite/Funds+Deposit+ListViewController
Lite/Funds+Deposit+DetailViewController

Lite/Funds+Deposit+View

Lite/Funds+Deposit+Cell

Lite/Funds+Deposit+Model 仅匹配最外层

General File Name: ModuleName+ XXXs.swift

- BinanceCapitalModuleABTests.swift
- BinanceCapitalModuleAnalytics.swift
- BinanceCapitalModuleGadgetKeys.swift
- BinanceCapitalModuleTypes.swift
- BinanceCapitalModuleURLs.swift

Class 访问控制：

Access control

Proper use of access control will greatly increase the**robustness**and**testability**of our code, especially in the context of modular architecture.

Use***private/fileprivate***to encapsulate declarations that are not supposed to be seen by external codes. Use**private**over**fileprivate**unless complained by compiler.

Use***final***for declarations that are not supposed to be inherited.

Use***public***for declarations that are supposed to be used outside of the current module

Use**open**for classes/functions that are supposed to be inherited somewhere else. In such case we should include some comments to clarify and gives directions to the use of the class/function.

Omit**internal**when they' re the default level, unless the compiler complains.

if/else/guard/early return/switch:

if return /guard

if/else 尽量不要超过2层 Avoid nesting too many `if else`

use map/reduce instead of For in

尽量使用高阶函数代替简单的for

Don't use magic value (number, string):

A "magic value" is a string or number used in a program that is essential to its proper function but provides no context or explanation for why it is what it is

example :

```
view.layer.cornerRadius = 4  //magic number

textLabel.text = "controllers.funds.overviewFundsViewController.tip".localized //magic string
```

replace:

```
access control enum Constants {

    static  let radius = 4

    static let textString :String { "controllers.funds.overviewFundsViewController.tip".localized }

}

view.layer.cornerRadius = Constants.radius

textLabel.text = Constants.textString
```

Split views/cells/state into separate files:

don't put classes such as views and cells in viewcontroller.swift file

iOS Team Portal Code Standards: https://git.toolsfdg.net/fe/iOS-Team-Portal/blob/master/Standards/Code%20Standards.md

Google Swift Guide: https://pages.swift.gg/google-swift-style-guide-in-chinese/

Swift Api Design Guide lines EN: https://www.swift.org/documentation/api-design-guidelines/

Swift Api Design Guide lines CN: https://github.com/SketchK/the-swift-api-design-guidelines-in-chinese