Target-Action命令行使用说明

1.概述

Target-Action组件化需要产生一定的硬编码，而这种硬编码可以按照文档规范约束。既然有规范，可以通过脚本自动产生约束的所有代码

2.约束内容介绍

2.1 Category工程代码约束

- 工程约定工程名字为：模块名+Module_Category。比如SaleModule_Category
- 分类名字为：模块名+Module。比如CTMediator (SaleModule)
- 对其他模块公共的接口均加上“模块名_”前缀。比如：

```
-(UIViewController *)sale_goodsDetailViewController:(nullable NSString *)goodId ;
```

- 函数字典中参数的key值定义为：k+模块名+Param+参数。比如：

```
NSString *const kSaleParamGoodId = @"goodid";
NSString *const kSaleParamDidlaunchOptions = @"didlaunchoptions";
NSString *const kSaleParamSex = @"sex";
```

- 对其他模块服务实体，全部转为字典，字典的key定义为：k+模块名+Entity+参数。比如：

```
pragma mark - GoodsModel Entiry Dic key
NSString *const kSaleEntityGoodsId = @"goodsid";
NSString *const kSaleEntityName = @"name";
```

2.2 服务模块的Target-Action代码约束

- 类名约束为Target*模块名+Module。比如Target*SaleModule
- 方法名约定Action_开头。比如：

```
-(UIViewController *)Action_goodsDetailViewController:(NSDictionary *)params;
```

3. 工具介绍

3.1 产生新的模块和Cagegory工程

运行Script目录下的createModule.sh和createModuleCategory.sh可分别快速产生一个新模块及对应的Category工程。

- ./createModule.sh 模块名
- ./reateModuleCategory.sh 模块名

一般不需要单独调用这两个命令，除非需要单独产生某一个工程

3.2 target_action_build.swift 介绍

target_action_build提供在设计号模块接口后，自动产生Category工程及模块工程以及部分约定代码的产生

3.2.1 编译swift文件为可执行文件

- xcrun swiftc target_action_build.swift

执行后产生target_action_build可执行文件

3.2.2 target_action_build可使用参数介绍

- -m参数，指模块名。为必选参数
- -i参数，本模块对外提供接口声明.h文件。可选参数。和-e必须存在一个。如果没有提供-scpath参数，新产生的接口会写入名为：模块+ModuleOutNoProj产生：模块名+ModuleCategory.{h,m}； Target_模块名+Module.{h,m}
- -e参数，外部模块需要使用本模块的实体声明.h文件。可选参数。和-i必须存在一个。如果没有提供-scpath参数，实体定义会写入当前名为：模块+ModuleOutNoProj产生：模块名+ModuleCategory.{h,m}
- -scpath Script文件夹所在路径，可选参数。如果存在会自动产生模块工程和Category工程以及产生大部分代码；如果不存在参见-e和-i解释，需要手动将模块名+ModuleCategory.{h,m}、 Target_模块名+Module.{h,m}文件内容copy到对应的工程文件

3.2.3 命令行使用Demo

假如存在名为Goods模块，其对外提供的接口在GoodsService.h文件；且其他模块依赖Goods模块的实体定义在GoodsModel.h中。

依次运行以下两个命令：

- xcrun swiftc target*action*build.swift
- ./target*action*build -m Goods -i Demo/GoodsService.h -scpath ./Script -e Demo/GoodsModel.h

执行后会在GoodsModuleOut文件夹下产生Goods和GoodsModuleCategory工程；且Category工程已自动完成代码生成不需要做任何更改；而Goods工程只需要补全业务逻辑即可。

假如不指定-scpath参数：

- ./target*action*build -m Goods -i Demo/GoodsService.h -e Demo/GoodsModel.h

自动产生的代码会自动写入到GoodsModuleOutNoProj文件下对应的文件中。需要手动将代码考到对应的Category工程和模块工程中