

UPYUN 架构与运维大会 Rediscover Arch&Ops

2015.11.28 北京

100offer

高薪招聘平台

告别盲目简历

让海量好机会主动来找你

首页 > iOS开发

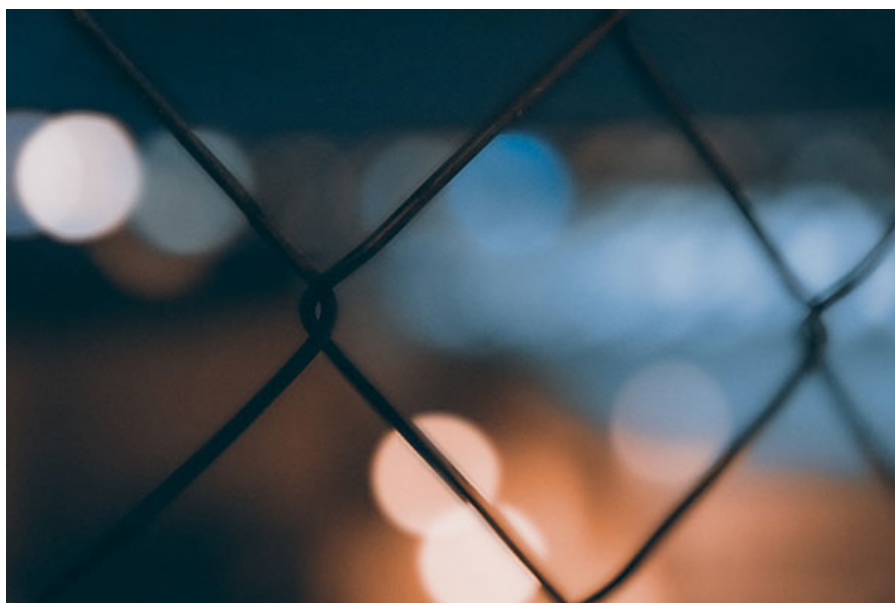
接口编程那些事（或者面向协议编程）

2015-12-14 06:18 编辑: lansekuangtu 分类: iOS开发 来源: 庞海礁 投稿

0 237

接口编程

招聘信息: Android开发工程师



接口是一系列可调用方法的集合。何为接口编程？接口编程是指当写一个函数或一个方法时，我们应该更加关注具体的接口，而不是实现类。具体理解可以参考这篇[文章](#)

在OC中，接口又可以理解为Protocol，面向接口编程又可以理解为面向Protocol编程，或者面向协议编程。在Swift中，苹果大幅强化了Protocol在这门语言中的地位，整个Swift标准库也是基于Protocol来设计的，有兴趣的童鞋可以看看这篇[文章](#)。面向接口编程正逐步成为程序开发的主流思想

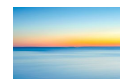
在实际开发中，大多数朋友都比较熟悉对象编程，比如，使用ASIHttpRequest执行网络请求

```
1 ASIHttpRequest *request = [ASIHttpRequest requestWithURL:url];
2 [request setDidFinishSelector:@selector(requestDone:)];
3 [request setDidFailSelector:@selector(requestWrong:)];
4 [request startAsynchronous];
```

request是请求对象，当发起请求时，调用者需要知道给对象赋哪些属性或者调用对象哪些方法。然而，使用AFNetworking请求方式却不尽相同

```
1 AFHTTPRequestOperationManager *manager = [AFHTTPRequestOperationManager manager]
2 [manager GET:@"www.olinone.com" parameters:nil success:^(AFHTTPRequestOperation
```

热门资讯



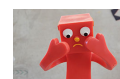
如何让iOS保持界面流畅？这些技巧你知道吗

点击量 13120



如何设计一个iOS控件？(iOS控件完全解析)

点击量 12307



培训机构毕业的程序员被歧视的背后逻辑

点击量 7169



苹果Swift编程语言开源 有望进一步普及

点击量 6210



对Xcode菜单选项的详细探索（干货）

点击量 6152



【资源集合】94个iOS开发资源推荐，帮你加

点击量 6072



Swift 3 API设计准则

点击量 5504



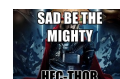
一款Loading动画的实现思路（一）：拆分复

点击量 5411



iOS事件处理机制与图像渲染过程

点击量 5358



Swift中编写单例的正确方式

点击量 4913

综合评论

mark

wooshare 评论了 【资源集合】94个iOS开发资源推荐，帮你加速应用...

mark

youkini 评论了 【资源集合】94个iOS开发资源推荐，帮你加速应用...

6666666

HAlice 评论了 深入理解RunLoop

mark

2yscw 评论了 快速搭建一个成熟，强

```
3     NSLog(@"好网站，赞一个！");
4 } failure:^(AFHTTPRequestOperation *operation, NSError *error) {
5     //to do
6 }];
```

同是请求对象，使用AFNetworking发起请求时，调用者可以不需要关心它有哪些属性，只有接口无法满足需求时才需要了解相关属性的定义。两种设计思路完全不同，当然，此处并不是想表明孰优孰劣，只是想通过两种截然不同的请求方式引出本文的思想——面向接口编程（或者面向协议编程）

接口比属性直观

在对象编程中，定义一个对象时，往往需要为其定义各种属性。比如，ReactiveCocoa中RACSubscriber对象定义如下

```
1 @interface RACSubscriber ()
2
3 @property (nonatomic, copy) void (^next)(id value);
4 @property (nonatomic, copy) void (^error)(NSError *error);
5 @property (nonatomic, copy) void (^completed)(void);
6
7 @end
```

参考AFNetworking的思想，以接口的形式提供访问入口

```
1 @interface RACSubscriber
2
3 + (instancetype)subscriberWithNext:(void (^)(id x))next
4                               error:(void (^)(NSError *error))error
5                               completed:(void (^)(void))completed;
6
7 @end
```

通过接口的定义，调用者可以忽略对象的属性，聚焦于其提供的接口和功能上。程序猿在首次接触陌生的某个对象时，接口往往比属性更加直观明了，抽象接口往往比定义属性更能描述想做的事情

接口依赖

设计一个APIService对象

```
1 @interface ApiService : NSObject
2
3 @property (nonatomic, strong) NSURL *url;
4 @property (nonatomic, strong) NSDictionary *param;
5
6 - (void)execNetRequest;
7
8 @end
```

正常发起Service请求时，调用者需要直接依赖该对象，起不到解耦的目的。当业务变动需要重构该对象时，所有引用该对象的地方都需要改动。如何做到既能满足业务又能兼容变化？抽象接口也许是一个不错的选择，以接口依赖的方式取代对象依赖，改造代码如下

```
1 @protocol ApiServiceProtocol
2 - (void)requestNetWithUrl:(NSURL *)url Param:(NSDictionary *)param;
3
4 @end
5
6 @interface NSObject (ApiServiceProtocol)
7 @end
8
9 @implementation NSObject (ApiServiceProtocol)
10
11 - (void)requestNetWithUrl:(NSURL *)url Param:(NSDictionary *)param {
12     ApiService *apiSrevice = [ApiService new];
13     apiSrevice.url = url;
14     apiSrevice.param = param;
15     [apiSrevice execNetRequest];
16 }
17
18 @end
```

壮的App框架...

mark

showhill 评论了 【资源集合】94个iOS开发资源推荐，帮你加速应用...

看完之后新手会有新手的收获 老手会有老手的收获 真正不屑一顾是 处在中间 452530253 评论了 编程语言大牛王垠：编程的智慧，带你少走弯路...

mark

1778031092 评论了 【资源集合】94个iOS开发资源推荐，帮你加速应用...

mark

lazydo 评论了 【资源集合】94个iOS开发资源推荐，帮你加速应用...

mark

魔鬼分界线 评论了 【资源集合】94个iOS开发资源推荐，帮你加速应用...

我擦，5点下班

feifei_Smile 评论了 我在 Google 做技术经理的一天...

相关帖子

box2d中的旋转关节为什么要有两个物体

能帮我推荐一个swift开发的书籍么

iOS8和ios9的适配，有哪些好的书籍和博客推荐(◕ ◕ ◕)啊！

推送跳转页面

请教各位，医疗类、生活类，畅销榜第十名、第二十名分别收入是多少？

iOS json 解析

微信授权问题

123

storyboard中这个弄没了怎么办啊

通过接口的定义，调用者可以不再关心ApiService对象，也无需了解其有哪些属性。即使需要重构替换新的对象，调用逻辑也不受任何影响。调用接口往往比访问对象属性更加稳定可靠

抽象对象

定义ApiServiceProtocol可以隐藏ApiService对象，但是受限于ApiService对象的存在，业务需求发生变化时，仍然需要修改ApiService逻辑代码。如何实现在不修改已有ApiService业务代码的条件下满足新的业务需求？

参考Swift抽象协议的设计理念，可以使用Protocol抽象对象，毕竟调用者也不关心具体实现类。Protocol可以定义方法，可是属性的问题怎么解决？此时，装饰器模式也许正好可以解决该问题，让我们试着继续改造ApiService

```
1  @protocol ApiService
2  // private functions
3
4  @end
5
6  @interface ApiServicePassthrough : NSObject
7
8  @property (nonatomic, strong) NSURL *url;
9  @property (nonatomic, strong) NSDictionary *param;
10
11  - (instancetype)initWithApiService:(id)apiService;
12  - (void)execNetRequest;
13
14  @end
15
16
17  @interface ApiServicePassthrough ()
18
19  @property (nonatomic, strong) id apiService;
20
21  @end
22
23  @implementation ApiServicePassthrough
24
25  - (instancetype)initWithApiService:(id)apiService {
26      if (self = [super init]) {
27          self.apiService = apiService;
28      }
29      return self;
30  }
31
32  - (void)execNetRequest {
33      [self.apiService requestNetWithURL:self.url Param:self.param];
34  }
35
36  @end
```

经过Protocol的改造，ApiService对象化身为ApiService接口，其不再依赖于任何对象，做到了真正的接口依赖取代对象依赖，具有更强的业务兼容性

定义一个Get请求对象

```
1  @interface GetApiService : NSObject
2  @end
3
4  @implementation GetApiService
5
6  - (void)requestNetWithURL:(NSURL *)url Param:(NSDictionary *)param {
7      // to do
8  }
9
10 @end
```

请求代码也随之改变

```
1  @implementation NSObject (ApiServiceProtocol)
2
3  - (void)requestNetWithURL:(NSURL *)url Param:(NSDictionary *)param {
4      id apiService = [GetApiService new];
5      ApiServicePassthrough *apiServicePassthrough = [[ApiServicePassthrough alloc] initWithApiService:apiService];
6      apiServicePassthrough.url = url;
7      apiServicePassthrough.param = param;
8      [apiServicePassthrough execNetRequest];
9  }
10
```

微博



CocoaChina

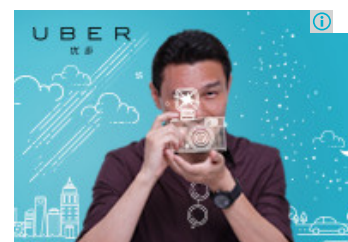
已关注

【经验之谈！35岁程序员的独家面试经历】创业失败后，在找工作。选择了三家（两家上市公司，一家将上市），都走到了关键的节点。我记录了面试过程中被问到的一些问题，希望对自己将来的面试有帮助，也希望对别人有所启发。http://t.cn/R4z5H48



12月11日 21:00 转发(16) | 评论(4)

【源码推荐(12.11)】①多种格式的collection流与scrollview联动结合demo
②系统原生二维码扫描③DropDownDemo，以上源码下载：http://t.cn/R4z5HJ1



顺路接一单, 有人聊天找灵感

加入我们, 还能赚油费 >

```
11 | @end
```

对象可以继承对象，Protocol也可以继承Protocol，并且可以继承多个Protocol，Protocol具有更强的灵活性。某一天，业务需求变更需要用到新的Post请求时，可以不用修改 GetApiService一行代码，定义一个新的 PostApiService实现Post请求即可，避免了对对象里面出现过多的if-else代码，也保证了代码的整洁性

依赖注入

文章写到这里，细心的童鞋可能已经发现问题——GetApiService依然是以对象依赖的形式存在。如何解决这个问题？没错，那就是依赖注入！

依赖注入是什么？借用[博客](#)里面的一句话，其最大的特点就是：帮助我们开发出松散耦合、可维护、可测试的代码和程序。这条原则的做法是大家熟知的面向接口，或者说是面向抽象编程。[objc](#)上这篇文章介绍的不错，有兴趣的童鞋也可以看看，在此就不再累述

基于依赖注入[objection](#)开源库的基础上继续改造ApiService

```
1 | @implementation NSObject (ApiServiceProtocol)
2 |
3 | - (void)requestNetWithUrl:(NSURL *)url Param:(NSDictionary *)param {
4 |     id apiSrevice = [[JSObjection createInjector] getObject:[GetApiService clas
5 |     ApiServicePassthrough *apiServicePassthrough = [[ApiServicePassthrough allo
6 |     apiServicePassthrough.url = url;
7 |     apiServicePassthrough.param = param;
8 |     [apiServicePassthrough execNetRequest];
9 | }
10 |
11 | @end
```

调用者关心请求接口，实现者关心需要实现的接口，各司其职，互不干涉

接口和实现分离的设计适用于团队协作开发，实现了系统的松散耦合，便于以后升级扩展。当然，接口编程对开发人员的要求也比较高，需要提前定义好接口，接口一变，全部乱套，这就是所谓的设计比实现难，但是设计接口的人工资都高啊！！

后记：你可以在[github](#)找到我，也可以通过[微博](#)联系我，感谢你的来访！



微信扫一扫

订阅每日移动开发及APP推广热点资讯

公众号：CocoaChina

我要投稿

收藏文章

分享到：

上一篇：换一个姿势写界面样式

下一篇：源码推荐(12.11B)：视图动画，仿SegmentFault

相关资讯



我来说两句



你怎么看？快来评论一下吧！

发表评论

所有评论 (0)

[关于我们](#) [商务合作](#) [联系我们](#) [合作伙伴](#)

北京触控科技有限公司版权所有

©2015 Chukong Technologies, Inc.

京ICP备 11006519号 京ICP证 100954号 京公网安备11010502020289  京网文[2012]0426-138号