

[iOS开发](#) [App Store研究](#) [应用评测](#) [产品设计](#) [Cocos引擎](#) [WebApp](#) [Swift](#) [游戏开发](#) [苹果相关](#) [营销推广](#) [业界动态](#) [程序人生](#)[首页](#) > [iOS开发](#)

寒哥教你学 iOS - 经验漫谈

2015-09-07 09:06 编辑: lansekuangtu 分类: iOS开发 来源: 南栀倾寒投稿

0 3859

[NSNumber](#) [iOS](#) [load](#)

招聘信息: iOS资深工程师



本篇文章主要讲解 4个问题

1. load妙用
2. aop面向切面编程
3. NSNumber Or Int
4. @()适配64位

1 让AppDelegate 减少负担



经过长时间的学习 你终于掌握了iOS大法 你找到了份iOS开发的工作 信誓旦旦的要开始你的coding生涯 老板对你非常器重 然后告诉你 我觉得你的技术 是非常刁的 那这个项目就你自己来搞吧 啊哦这就意味着这个项目你就从头到尾处理 从软件的架构 到页面的展示 都交给你喽??

用着自己的半吊子水平 papapa的 coding 决心一定要把代码封装好 写的漂亮 (其实是听大神说 封装 其实自己不太懂)

项目到了尾声 老板告诉你我们的app 我们的app 将来得来个分享到朋友圈的功能吧 不然怎么体现我产品的牛逼

然后你听说友盟比较好使 (有广告的嫌疑) 你去友盟看了他们的文档 他告诉你你要在 appDelegate didFinishLaunch 方法里面写了这个东西

热门资讯

[通过iOS 9
SFSafariViewController](#)
点击量 27308[从零开始学 iOS 开发的15条建议](#)
点击量 15954[iOS 万能跳转界面方法 \(runtime实用篇一\)](#)
点击量 13447[《招聘一个靠谱的 iOS》面试题参考答案](#)
点击量 11469[谁让App工程师产生了泡沫?](#)
点击量 10998[iOS 大型项目开发漫谈](#)
点击量 10002[iOS 面试大全从简单到复杂 \(简单篇\)](#)
点击量 8411[SQLite这么娇小可爱, 不多了解点都不行啊](#)
点击量 6931[你给我解析清楚, 都有了CALayer了, 为什么](#)
点击量 6734[iOS: 学习runtime的理解和心得](#)
点击量 6272

综合评论

方式“字阵”记录下来: 少叠测时。这是什么方法 求教呢
[Apple_MrPig](#) 评论了 [技术学习总原则](#)

谁发的发的说法是第三代
[dqw18037](#) 评论了 [集成新浪,腾讯,人人分享功能的Demo](#)

总结的比较全面, 一字一句看完了
[庸者的救赎](#) 评论了 [Swift学习: 从Objective-C到Swi...](#)

霸气
[阿花君霸占路人](#) 评论了 [独立游戏人: 像素风格游戏制作分享](#)

必须要有开发者证书吗
[CocosKing](#) 评论了 [Cocoapods 应用第](#)

```
[UMSocialData setAppKey:@"XX"];

//      注册微信

[UMSocialWechatHandler setWXAppId:@"XXX" appSecret:@"XX" url:@""];

//      注册QQ

[UMSocialQQHandler setQQWithAppId:@"XXX" appKey:@"XXX" url:@""];
```

过了几天 老板又说 我们需要统计下我页面的信息 你接入了友盟的统计 在appdelegate didFinishLaunch又 多了行代码

需求是无穷无尽 我需要bug统计（fir hud） 提醒用户评分系统（iRate） 推送（jPush 信鸽 个推。。）

当初你决心一定要把代码封装的完美 写的漂亮的心早就被老板的需求彻底打败了

别担心 寒哥教你小技巧

不知道你们用过 **IQKeyBoardManage**和**iRate**这种智能库没

大牛的readme 写了这段话

Key Features

- 1) CODELESS, Zero Line Of Code 不需要写任何代码
- 2) Works Automatically //自动工作
- 3) No More UIScrollView //不需要scrollview
- 4) No More Subclasses //不需要继承父类
- 5) No More Manual Work //不需要配置
- 6) No More #imports //不需要导入

其实不神奇 只是大牛用了 + load这个方法

学习OC都知道这个代码会在一个类被加载到运行库中就会被自动调用 这不就实现了 自动调用

写一个类继承自NSObject

```
#import @interface ThirdPartService : NSObject@end

@implementation ThirdPartService

+ (void)load {static dispatch_once_t onceToken;dispatch_once(&onceToken, ^{      //      TODO  这里是我

    [FIR handleCrashWithKey:@"XX"];      //      友盟

    [UMSocialData setAppKey:@"XX"];      //      隐藏未安装的平台

    [UMSocialConfig hiddenNotInstallPlatforms:@[UMShareToQQ,UMShareToQzone,UMShareToWechatSession,U

    [UMSocialWechatHandler setWXAppId:@"XX" appSecret:@"XX" url:@""];      //      注册QQ
```

一部分 - Xcode ...

WebView.JavascriptBridge很好用，就是webview如果在iOS7(uiwebview)上和那爱离殇 评论了 iOS7之后JavaScript与Objectiv...

一看就知道小编不是程序员，百度永远代替不了googol。不过就算回归了，估NearKong 评论了 重返中国大陆？你好啊谷歌

CocoaChina真是什么文章都放啊 继续前行 评论了 一次 TableView 性能优化经历

屌爆了 牛人 夹脚鞋走遍世界 评论了 【哔哔哔】加班汪是怎样炼成的？

学习学习 yangchong5477 评论了 【实例教程】你会用swift创建复杂的加载动画吗

相关帖子

在iOS中使用了pods，但在引入头文件的时候没有代码提示，该怎么办？

button控制pageviewController自动翻页

用Storyboard的时候如何添加超过显示界面的东西啊

我是笨鸟：想问关于网络软件问题

quick 如何添加library项目

xcode 快捷键恢复默认

ios

为什么真机上与模拟器尺寸不符？

cocos2d-x lua 使用pluginx接入ios的 appstore付费，有什么文档可以参考的

```
// TODO QQ的不是真的

[UMSocialQQHandler setQQWithAppId:@"XX" appKey:@"XX" url:@""]; // TODO UM统计

[MobClick startWithAppkey:@""];

[MobClick setCrashReportEnabled:NO]; NSLog(@"第三方服务注册完毕");

});

@end
```

类似于定位也可以这样写

```
5 // Created by nero on 15/8/3.
6 // Copyright (c) 2015年 nero. All rights reserved.
7
8 #import "CLLocationManager.h"
9 #import "FFCommonDefine.h"
10 @interface CCLocationManager ()<UIAlertViewDelegate>{
11     CLLocationManager *_manager;
12 }
13
14 @property (nonatomic, strong) CLLocationManager *manager;
15 @property (nonatomic, strong) LocationBlock locationBlock;
16 @property (nonatomic, strong) NSStringBlock cityBlock;
17 @property (nonatomic, strong) NSStringBlock addressBlock;
18 @property (nonatomic, strong) LocationErrorBlock errorBlock;
19
20 @end
21
22 @implementation CCLocationManager
23
24 + (void)load {
25     // 这里自己调用，因为runtime 会帮我们调用
26     [super load];
27     static dispatch_once_t onceToken;
28     dispatch_once(&onceToken, ^{
29         CLLocationManager *manager = [self shareLocation];
30         manager.manager = [[CLLocationManager alloc] init];
31
32         // [UIApplication sharedApplication].idleTimerDisabled = YES;
33         _manager = [[CLLocationManager alloc] init];
34         [_manager requestAlwaysAuthorization]; //NSLocationAlwaysUsageDescription
35
36         if ([manager.manager respondsToSelector:@selector(requestWhenInUseAuthorization)]) {
37             [manager.manager requestWhenInUseAuthorization]; //NSLocationWhenInUseDescription
38         }
39
40
41     });
42 }
43
44 }
```

模块和服务完全拆开

但是有的服务 如APNS需要LaunchOption 那就只能写在appDelegate 不过这样的话已经摘除很多代码了 只剩下几个固定的 到时候再修改appDelegate就会感觉非常清晰 了

2 ViewController继承?

接着上面讲 我们接入了友盟统计 友盟统计最基本的东西就是 统计页面的pv

```
89 @param second 单位:秒, 最小90秒, 最大604800秒(24hour).
90 @return void.
91
92 + (void)setLogSendInterval:(double)second;
93
94 /** 设置日志延迟发送
95  @param second 设置一个 [0, second] 范围的延迟发送秒数, 最大值1800s.
96  @return void
97  */
98 + (void)setLatency:(int)second;
99
100 ///
101 /// @name 页面计时
102 ///
103
104 /** 手动页面时长统计, 记录某个页面展示的时长.
105  @param pageName 统计的页面名称.
106  @param seconds 单位为秒, int型.
107  @return void.
108  */
109 + (void)logPageView:(NSString *)pageName seconds:(int)seconds;
110
111 /** 自动页面时长统计, 开始记录某个页面展示时长.
112  使用方法: 必须配对调用beginLogPageView:和endLogPageView:两个函数来完成自动统计, 若只调用某一个函数不会生成有效数据.
113  在该页面展示时调用beginLogPageView:, 当退出该页面时调用endLogPageView:
114  @param pageName 统计的页面名称.
115  @return void.
116  */
117 + (void)beginLogPageView:(NSString *)pageName;
118
119 /** 自动页面时长统计, 结束记录某个页面展示时长.
120  使用方法: 必须配对调用beginLogPageView:和endLogPageView:两个函数来完成自动统计, 若只调用某一个函数不会生成有效数据.
121  在该页面展示时调用beginLogPageView:, 当退出该页面时调用endLogPageView:
122  @param pageName 统计的页面名称.
123  @return void.
124  */
125 + (void)endLogPageView:(NSString *)pageName;
126
127 #pragma mark event logs
```

友盟的这样写 对于新手的我们就觉得这不就so easy吗

我打开了某个vc (HomeController)

在代码里面写上了这句

```
-(void)viewWillAppear:(BOOL)animated {
    [super viewWillAppear:animated];#ifndef DEBUG
    [MobClick beginLogPageView:NSStringFromClass([self class])];#endif}

-(void)viewWillDisappear:(BOOL)animated {
    [super viewWillDisappear:animated];#ifndef DEBUG

    [MobClick endLogPageView:NSStringFromClass([self class])];#endif}
```

然后我一个项目中可能有几十个 甚至上百个页面需要统计pv 我总不能每个节目都这样写吧

聪明的我们想到了继承

如MyBaseViewController:UIViewController

这样就要做一件事 把我们项目中所有继承自UIViewController的类全部改为继承自MyBaseViewController 然而你真的觉得这样好吗 我们一个项目中有几十个控制器 我就要把每个控制器改一遍

这种重复性的工作一是无聊 而是容易出错 你复制着复制者就会遗漏掉某个类 重要的是 我们项目中很多类并不是直接继承自UIViewController 有的可能是UITableViewController UICollectionViewContr0ller UINavigationController 甚至不常用的UISearchDisplayController UIPopoverController ?UIPresentController 是不是突然觉得这么多啊啊 ??

这也不是坑的 坑的是将来你混成了大牛 招了个小弟 你告诉他你所有的类都要继承自我写的各种父类 新手总是会不经意间犯错误 有些类忘记继承了 后期查起来难度非常大 浪费时间 所以这种设计是不合理的

- 寒哥再次教你黑魔法 Method swizzling

关于这个是干嘛的 自行百度

这里有一篇来自NSHipster博主的文章 [英文](#)

[中文翻译](#)

还有一篇解释runtime的文章[传送门](#)

[实践](#)

用方法交叉 我们就可以拦截吸引的方法了 上代码了

这样就做到了面向切面编程 (AOP) 的思想

上代码

```
#import @interface UIViewController (AOP)#warning 运行时 改变一下方法 做一些切面编程比如 统计 等等
@end

#import "UIViewController+AOP.h"
```

```
#import #import

@implementation UIViewController (AOP)

+ (void)load {    static dispatch_once_t onceToken;    dispatch_once(&onceToken, ^{
    Class class = [self class];    // When swizzling a class method, use the following:
    // Class class = object_getClass((id)self);
    swizzleMethod(class, @selector(viewDidLoad), @selector(aop_viewDidLoad));
    swizzleMethod(class, @selector(viewDidAppear:), @selector(aop_viewDidAppear:));
    swizzleMethod(class, @selector(viewWillAppear:), @selector(aop_viewWillAppear:));
    swizzleMethod(class, @selector(viewWillDisappear:), @selector(aop_viewWillDisappear:));
});
} void swizzleMethod(Class class, SEL originalSelector, SEL swizzledSelector) {
Method originalMethod = class_getInstanceMethod(class, originalSelector);
Method swizzledMethod = class_getInstanceMethod(class, swizzledSelector);BOOL didAddMethod =
class_addMethod(class,
                originalSelector,
                method_getImplementation(swizzledMethod),
                method_getTypeEncoding(swizzledMethod));if (didAddMethod) {
class_replaceMethod(class,
                    swizzledSelector,
                    method_getImplementation(originalMethod),
                    method_getTypeEncoding(originalMethod));
} else {
    method_exchangeImplementations(originalMethod, swizzledMethod);
}
}

- (void)aop_viewDidAppear:(BOOL)animated {
[self aop_viewDidAppear:animated];
}

-(void)aop_viewWillAppear:(BOOL)animated {
[self aop_viewWillAppear:animated];#ifndef DEBUG
    [MobClick beginLogPageView:NSStringFromClass([self class])];#endif
-(void)aop_viewWillDisappear:(BOOL)animated {
    [self aop_viewWillDisappear:animated];#ifndef DEBUG

    [MobClick endLogPageView:NSStringFromClass([self class])];#endif
- (void)aop_viewDidLoad {
[self aop_viewDidLoad];if ([self isKindOfClass:[UINavigationController class]]) {    UINavigationController
    nav.navigationBar.translucent = NO;
    nav.navigationBar.barTintColor = GLOBAL_NAVIGATION_BAR_TIN_COLOR;
    nav.navigationBar.tintColor = [UIColor whiteColor];    NSDictionary *titleAtt = @{NSForegroundColor
    [[UINavigationController appearance] setTitleTextAttributes:titleAtt];
    [[UIBarButtonItem appearance]
```



```
setBackButtonTitlePositionAdjustment:UIOffsetMake(0, -60)

forBarMetrics:UIBarMetricsDefault];

} // self.view.backgroundColor = [UIColor whiteColor];self.navigationController.interactivePopGestures
} @end
```

图片代码一份 方便观看

```
13 @implementation UIViewController (AOP)
14
15 + (void)load {
16     static dispatch_once_t onceToken;
17     dispatch_once(&onceToken, ^{
18         Class class = [self class];
19         // When swizzling a class method, use the following:
20         // Class class = object_getClass((id)self);
21         swizzleMethod(class, @selector(viewDidLoad), @selector(aop_viewDidLoad));
22         swizzleMethod(class, @selector(viewDidAppear:), @selector(aop_viewDidAppear:));
23         swizzleMethod(class, @selector(viewWillAppear:), @selector(aop_viewWillAppear:));
24         swizzleMethod(class, @selector(viewWillDisappear:), @selector(aop_viewWillDisappear:));
25     });
26 }
27
28 void swizzleMethod(Class class, SEL originalSelector, SEL swizzledSelector) {
29     Method originalMethod = class_getInstanceMethod(class, originalSelector);
30     Method swizzledMethod = class_getInstanceMethod(class, swizzledSelector);
31     BOOL didAddMethod =
32     class_addMethod(class,
33                     originalSelector,
34                     method_getImplementation(swizzledMethod),
35                     method_getTypeEncoding(swizzledMethod));
36
37     if (didAddMethod) {
38         class_replaceMethod(class,
39                             swizzledSelector,
40                             method_getImplementation(originalMethod),
41                             method_getTypeEncoding(originalMethod));
42     } else {
43         method_exchangeImplementations(originalMethod, swizzledMethod);
44     }
45 }
46
47 -(void)aop_viewDidAppear:(BOOL)animated {
48     [self aop_viewDidAppear:animated];
49     // NSLog(@"3333333333");
50 }
51
52 -(void)aop_viewWillAppear:(BOOL)animated {
53     [self aop_viewWillAppear:animated];
54     // NSLog(@"2222222222");
55 }
56 #ifdef DEBUG
57 [MobClick beginLogPageView:NSStringFromClass([self class])];
58 #endif
59
60 -(void)aop_viewWillDisappear:(BOOL)animated {
61     [self aop_viewWillDisappear:animated];
62     #ifdef DEBUG
63     [MobClick endLogPageView:NSStringFromClass([self class])];
64 #endif
65 }
66
67 -(void)aop_viewDidLoad {
68     [self aop_viewDidLoad];
69     // NSLog(@"1111111111");
70     if ([self isKindOfClass:[UINavigationController class]]) {
71         UINavigationController *nav = (UINavigationController *)self;
72         nav.navigationBar.translucent = NO;
73         nav.navigationBar.barTintColor = GLOBAL_NAVIGATION_BAR_TIN_COLOR;
74         nav.navigationBar.tintColor = [UIColor whiteColor];
75         NSDictionary *titleAtt = @{NSForegroundColorAttributeName:[UIColor whiteColor]};
76         [[UINavigationController appearance] setTitleTextAttributes:titleAtt];
77         [[UIBarButtonItem appearance] setTitleTextAttributes:titleAtt];
78         setBackButtonTitlePositionAdjustment:UIOffsetMake(0, -60)
79         forBarMetrics:UIBarMetricsDefault];
80     }
81     // self.view.backgroundColor = [UIColor whiteColor];
82     self.navigationController.interactivePopGestureRecognizer.delegate = (id<UIGestureRecognizerDelegate>)self;
83 }
84 @end
85
```

我们充分利用了黑魔法达到了面向切面编程的好处

思想来源[这里http://casatwy.com/iosying-yong-jia-gou-tan-viewceng-de-zu-zhi-he-diao-yong-fang-an.html](http://casatwy.com/iosying-yong-jia-gou-tan-viewceng-de-zu-zhi-he-diao-yong-fang-an.html)

黑魔法非毒药 遵守一个规范写出来的代码是不会Crash的 只要能帮我们解决问题就是好东西

黑魔法性能 有瓶颈? 都到runtime的底层了 你还担心有瓶颈? 少年安心使用就好了? 不服 可以用Time Profiel测试

黑魔法也非万能? 像在导航控制器要封装手势 统一管理左侧返回按钮? 这些东西 还是继承来得好

技术就是工具 黑猫, 白猫, 抓住老鼠就是好猫

3 网络访问参数到底用基本数据类型还是对象

下面看两个方法

```

9
10 #import "ViewController.h"
11 #import <AFNetworking.h>
12 typedef void (^CompleteBlock)(id responseObject, NSError *error);
13 @interface ViewController ()
14 @end
15
16 @implementation ViewController
17
18 + (void)getDataAtPageNo:(NSNumber *)pageNo PageSize:(NSNumber *)pageSize complete:(CompleteBlock)complete {
19     NSMutableDictionary *param = [NSMutableDictionary dictionary];
20     if (pageSize) {
21         [param setObject:pageSize forKey:@"pageSize"];
22     }
23     [param setObject:pageNo forKey:@"pageNo"];
24     // SendRequest
25 }
26
27 + (void)getData2AtPageNo:(long )pageNo PageSize:(long )pageSize complete:(CompleteBlock)complete {
28     NSMutableDictionary *param = [NSMutableDictionary dictionary];
29     [param setObject:@(pageSize) forKey:@"pageSize"];
30     [param setObject:@(pageNo) forKey:@"pageNo"];
31     // SendRequest
32 }
33 @end
34

```

```

+ (void)getDataAtPageNo:(NSNumber *)pageNo PageSize:(NSNumber *)pageSize
complete:(CompleteBlock)complete {NSMutableDictionary *param = [NSMutableDictionary dictionary];

    [param setObject:pageSize forKey:@"pageSize"];

}

[param setObject:pageNo forKey:@"pageNo"];// SendRequest}

+ (void)getData2AtPageNo:(long )pageNo PageSize:(long )pageSize
complete:(CompleteBlock)complete {    NSMutableDictionary *param = [NSMutableDictionary dictionary];

    [param setObject:@(pageSize) forKey:@"pageSize"];

    [param setObject:@(pageNo) forKey:@"pageNo"];// SendRequest

}

```

在访问网络请求时 对于有参数的请求 设计一个方法 主流为以上两种

1. 使用对象当做参数
2. 使用基本数据类型做参数

一般情况下 这并没有什么大的区别 但是寒哥给出的意见是Never出现基本数据类型

一般情况下 开发者可能觉得并没有什么区别 下面我给大家举个例子

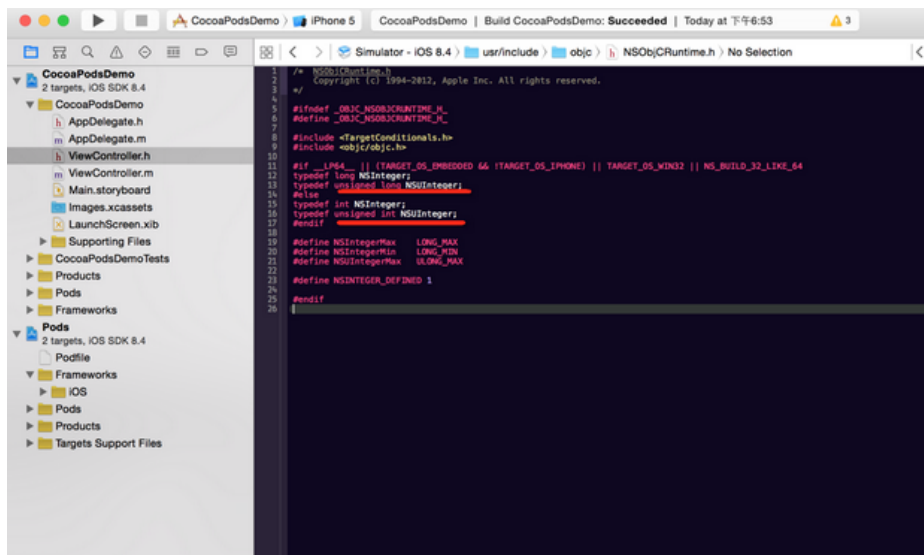
在设计一个分页展示数据的时候： 在页面上的逻辑就是 默认加载第一页 每页长度为10 (Server端的同学一般都很有友好 默认情况下 不传每页的长度就是10个) 但是传了就会覆盖掉后台写的默认参数 如传了20 Server就吐20条数据

1. 在第一中设计方案中： 可能在某个控制器中保留一个PageNo PageSize 的对象的成员变量， 在下拉刷新或者上拉加载的时候 会传递对应的参数给请求方法， 如果没有特殊需求的话pageSize 对象可有可无 也就是有可能为nil， 那在对应的param可能就没有这个参数传递给Server。 Server 就会交还给我们某页的20条数据
2. 在 第二种设计方案中： 可能也在某个控制器中保留一个PageNo pageSize的基本数据类型的成员变量， 在访问网络请求时交给对应的方法， 一般没有特殊需求我们也不会对PageSize专门设置 但是 基本数据类型在OC 和C语言这种传统的编程语言中是有默认值的 为0， 虽然我们没有给pageSize 赋值 但是默认系统默认给了0这个初始值 那么传递到Server的时候 就会覆盖掉Server 写的默认pageSize=10 这样的请求既不会报错 也不会返回数据

超级难调试

所以在网络访问中 寒哥给出的意见就是Never出现基本数据类型

4 用NSNumber比基本数据类型的好处？ 64位适配问题

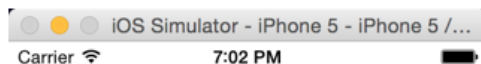


在32下设备是Int 在64位是long

我们都知道苹果不允许不支持64位的app上架 但是貌似我们从来没有为32位和64位做适配

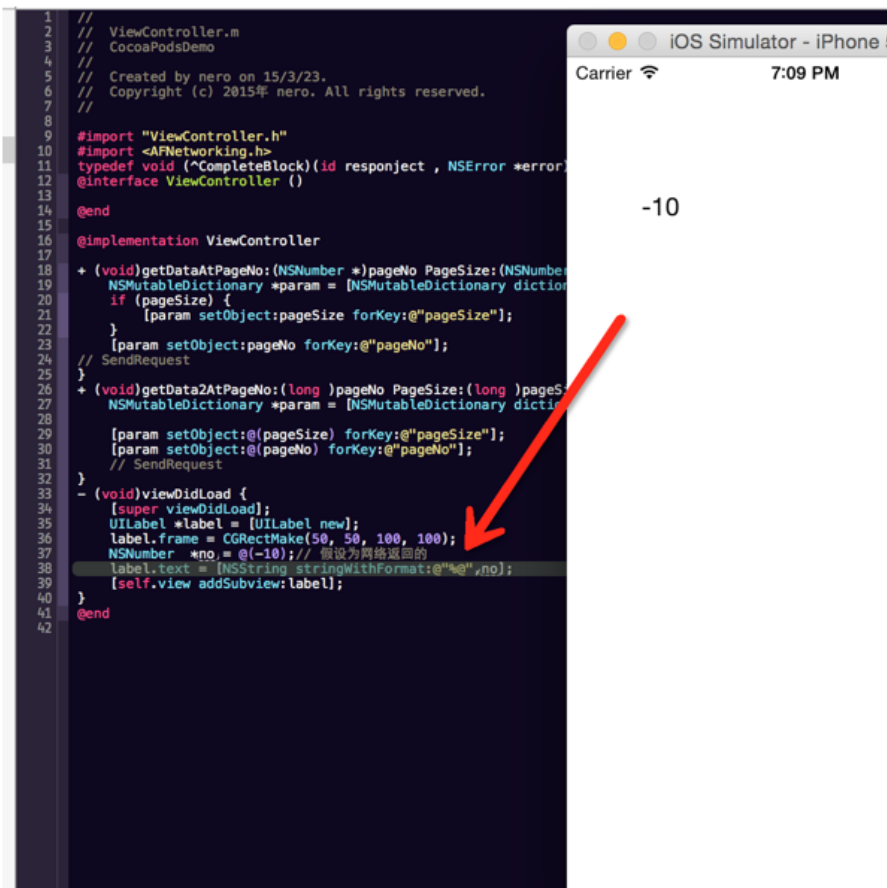
其实不然 在printf 和NSLog 时 对应%d %zd %f 占位符是非常严格的 如果不对项目就会造成意外的结果

```
32 /  
33 - (void)viewDidLoad {  
34     [super viewDidLoad];  
35     UILabel *label = [UILabel new];  
36     label.frame = CGRectMake(50, 50, 100, 100);  
37     NSNumber *no = @(-10); // 假设为网络返回的  
38     label.text = [NSString stringWithFormat:@"%u", [no unsignedIntegerValue]];  
39     [self.view addSubview:label];  
40 }
```



4294967286

其实拿到一个NSNumber我们并不知道他到底是int long unsigned int Bool 直接针对某个类型转换是有风险的 但是其实Clang 给我们提供了个非常好用的Macro @()



NSNumber并不是一个简单的类 它是cocoa 中 类簇的实现参考资料

<http://www.cocoachina.com/ios/20140109/7681.html>

<http://www.cocoachina.com/ios/20150106/10848.html>

<http://www.cocoachina.com/ios/20141218/10688.html>



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：Objective-C 与 Runtime：为什么是这样？

相关资讯	
iOS7之后JavaScript与Objective-C之间的通信	ReactiveCocoa2 源码浅析
源码推荐(9.06)：collectionView集合视图展开二级视图，	详解 CALayer 和 UIView 的区别和联系
iOS App创建桌面快捷方式	iOS 9 学习系列：MapKit 的新变化（Flyover、交通和自
自动化生成 APNS PEM 文件	Swift 开发 Uther 小记 - 一个简单的蠢萌机器人
【iOS】彩色TabBar切换动画实现	iOS开发之多图片无缝滚动组件封装与使用