Domob iOS SDK 4.x 使用手册

1 Demo 下载地址：

https://github.com/Domob-SDK/domob-ios-ad-sdk-sample
https://github.com/Domob-SDK/domob-ios-ad-sdk-cocos2dx-sample
https://github.com/Domob-SDK/domob-ios-ad-sdk-unity-plugin

2 必要条件

1. 使用 Xcode 5 或更高版本
2. 运行环境为 iOS 4.3 或更高版本

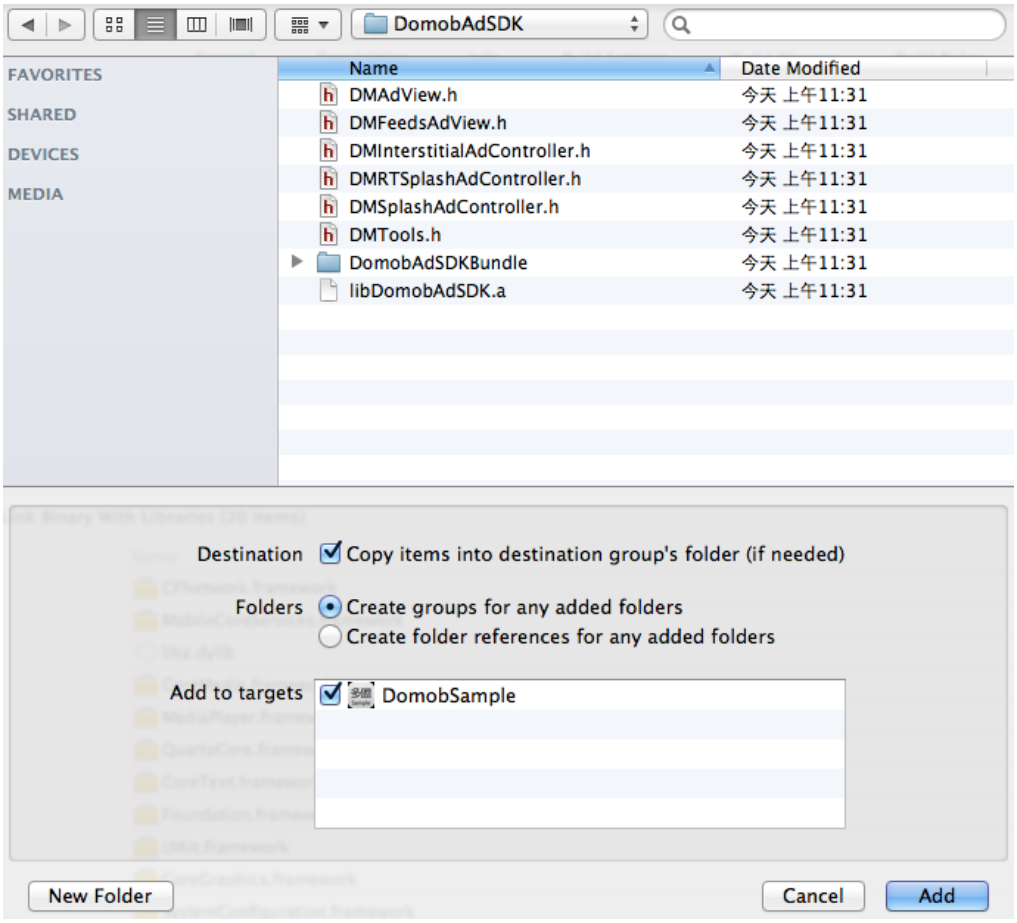
3 从多盟官网获取 PublisherID & PlacementID

如果想要在 App 中使用多盟的广告，首先需要获得一个 Publisher ID 和相应的 PlacementID。
请登陆多盟官网（http://www.domob.cn/help/index.htm#tab1），获取更多如何创建新用户以及新建应用的信息。
当新建应用并拥有一个PublisherID之后，添加广告位后即可获得广告位ID，如下图所示，详情请查看http://www.domob.cn/news/announcement/20130503/503.htm。

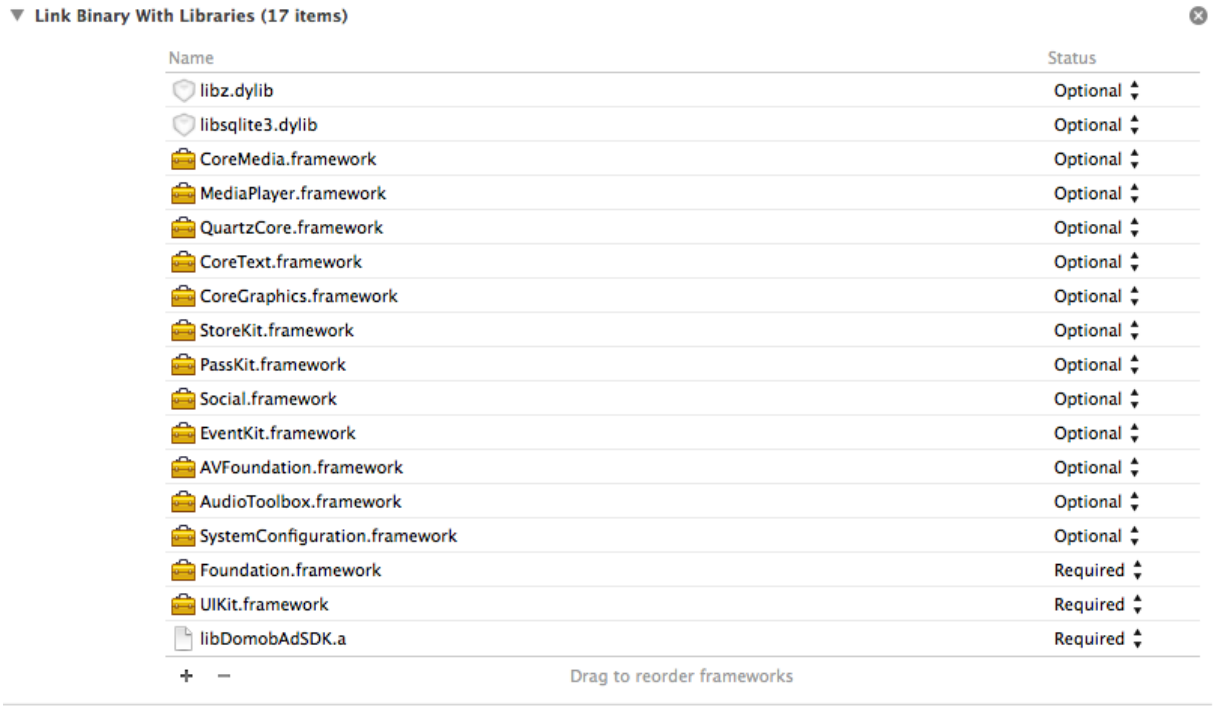
4 将 SDK 添加到项目中

SDK 的发行版本中需要添加到项目中的文件包括一个静态库、4 个头文件以及一个 Bundle 文件。

1. 将上述文件，添加到您的项目中。建议直接将“DomobAdSDK”文件夹添加到项目中。
包含的内容，如下图：



2. 若要 SDK 正常工作，需要向项目中添加下面这些 Framework，如下图：
(此处我们做了缩减)



3. 若要 SDK 正常工作，还需要在项目的 Build Settings → Linking → Other Linker Flags 添加 -ObjC， 如图:

5 广告视图的使用

5.1 Banner 广告的使用

（创建 Banner 广告位，请在官网“广告位管理”界面的“广告位类型设置中”请选择“inline 广告位”）

若要在应用中添加一个 Banner 广告，只需要简单的几步：

a. 导入 DMAdView.h 头文件。
b. 声明一个 DMAdView 的实例。
c. 使您的 UIViewController 实现 DMAdViewDelegate。
d. 在您的 UIViewController 的 m 文件中，使用您从多盟官网获得的 PublisherId 和 PlacementId 以及您期望的广告尺寸，来初始化 DMAdView。
e. 设置 DMAdView 的委派。
f. 设置 DMAdView 的 root view controller。
g. 设置 DMAdView 的 frame，把 DMAdView 放置在您希望的位置。
h. 将 DMAdView 添加到父视图中。
i. 最后，调用 loadAd 方法，开始请求广告。

下面的的代码片段，简要的描述了这个过程：

```
#import "DMAdView.h"

@interface DMBannerSampleViewController: UIViewController <DMAdViewDelegate>
```

```
{
    DMAdView *_dmAdView;
}
@end
```

```
@implementation DMBannerSampleViewController

- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
{
    self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
    if (self)
    {

        self.title = NSLocalizedString(@"Banner", @"Banner");

     // 创建广告视图，此处使用的是测试ID，请登陆多盟官网（www.domob.cn）获取新的ID
    // Creat advertisement view please get your own ID from domob website

    _dmAdView = [[DMAdView alloc] initWithPublisherId:@"56OJyGFYuMOI695Q87"
                                          placementId:@"16TLwMxaAc0izY7NJgmfgl5k"];

    // 设置广告视图的位置 宽与高设置为0即可 该广告视图默认是横竖屏自适应 但需要在旋转时调用
orientationChanged 方法
    // Set the frame of advertisement view
    _dmAdView.frame = CGRectMake(0, 20, FLEXIBLE_SIZE.width,FLEXIBLE_SIZE.height);
    }
    return self;
}

- (void)viewDidLoad
{
    [super viewDidLoad];


    _dmAdView.delegate = self; // 设置 Delegate
    _dmAdView.rootViewController = self; // 设置 RootViewController
    [self.view addSubview:_dmAdView]; // 将广告视图添加到父视图中
    [_dmAdView loadAd]; // 开始加载广告}

- (void)viewDidUnload
{
    [super viewDidUnload];
    [_dmAdView removeFromSuperview]; // 将广告试图从父视图中移除
}
//针对Banner的横竖屏自适应方法
//method For multible orientation
- (void)willRotateToInterfaceOrientation:(UIInterfaceOrientation)toInterfaceOrientation
                                duration:(NSTimeInterval)duration
{
    [_dmAdView orientationChanged];
}

- (void)dealloc
{
    _dmAdView.delegate = nil;
    _dmAdView.rootViewController = nil;
    [_dmAdView release];
    [super dealloc];
}
@end
```

Banner 默认是横竖屏自适应的，高度在手机上是 50，在平板上是 90，如果改用固定的宽度的 Banner 则需要使用 setAdSize 方法设置，创建过程如下：

```
    // Set the frame of advertisement view
```

```
    [_dmAdView setAdSize:DOMOB_AD_SIZE_320x50];
    _dmAdView.frame = CGRectMake(0, 20,
DOMOB_AD_SIZE_320x50.width,DOMOB_AD_SIZE_320x50.height);
```

关于 DMAdView 的释放：
在这里强烈建议您，在释放 DMAdView 之前，将 DMAdView 的属性 delegate 和 rootViewController 设置为 nil。

设置可选参数：
还可以为 DMAdView 设置一些可选参数，来提高广告投放的精准性。这些参数的设置方法都定义在 DMAdview.h 中，如下：

```
// 设置地理位置信息
- (void)setLocation:(CLLocation *)location;

// 设置邮编
- (void)setPostcode:(NSString *)postcode;

// 设置关键字
- (void)setKeywords:(NSString *)keywords;

// 设置用户年龄
- (void)setUserBirthday:(NSString *)userBirthday;

// 设置用户性别
- (void)setUserGender:(DMUserGenderType)userGender;
```

实现 DMAdViewDelegate：
您可以通过实现 DMAdViewDelegate 中定义的方法，来跟踪广告生命周期中的各个阶段。所有这些方法也都定义在 DMAdView.h 中，如下：

```
@protocol DMAdViewDelegate <NSObject>
@optional
// 成功加载广告后，回调该方法
- (void)dmAdViewSuccessToLoadAd:(DMAdView *)adView;
// 加载广告失败后，回调该方法
- (void)dmAdViewFailToLoadAd:(DMAdView *)adView withError:(NSError *)error;

// 当将要呈现出 Modal View 时，回调该方法。如打开内置浏览器。
- (void)dmWillPresentModalViewFromAd:(DMAdView *)adView;
// 当呈现的 Modal View 被关闭后，回调该方法。如内置浏览器被关闭。
- (void)dmDidDismissModalViewFromAd:(DMAdView *)adView;
// 当因用户的操作（如点击下载类广告，需要跳转到Store），需要离开当前应用时，回调该方法
- (void)dmApplicationWillEnterBackgroundFromAd:(DMAdView *)adView;

@end
```

5.2 插屏广告的使用

插屏广告的使用，同样非常简单，只需要几步即可完成。

a. 导入 DMInterstitialAdController.h 头文件。
b. 声明一个 DMInterstitialAdController 的实例。
c. 让您的 UIViewController 实现 DMInterstitialAdControllerDelegate 协议。
d. 在您的 Controller 的 m 文件中，使用 PublisherId 和 PlacementId 以及负责呈现插屏广告的 UIViewController 来初始化一个 DMInterstitialAdController。

e. 在合适的时候，调用 loadAd 方法，来预加载插屏广告。
f. 在想要呈现插屏广告的时候，调用 present 方法，来呈现插屏广告。
g. 在呈现完成后，需要重新通过 loadAd 方法，加载一条新广告，用于下次呈现。

下面的代码片段，简单描述了这个过程：

```
@interface DMInterstitialSampleViewController: UIViewController
<DMInterstitialAdControllerDelegate>
{
    DMInterstitialAdController *_dmInterstitial;
}
@end
```

```
@implementation DMInterstitialSampleViewController
- (void)viewDidLoad
{
    [super viewDidLoad];

    //  初始化插屏广告，此处使用的是测试ID，请登陆多盟官网（www.domob.cn）获取新的ID
    _dmInterstitial = [[DMInterstitialAdController alloc] initWithPublisherId:@"56OJyM1ouMGoULfJaL"
                                                                   placementId:@"16TLwebvAchkAY6iOWkE6kpk"
                                                            rootViewController:self
                                                                          size:DOMOB_AD_SIZE_300x250];

    // 设置插屏广告的Delegate
    _dmInterstitial.delegate = self;
    // 加载一条插屏广告
    [_dmInterstitial loadAd];

 }

- (void)viewDidUnload
{
    [super viewDidUnload];

    _dmInterstitial.delegate = nil; // 释放插屏广告对象前，请先将其delegate设置为nil
    [_dmInterstitial release]; // 释放插屏广告对象
}

- (void)onPresentBtnClicked:(id)sender
{
    // 在需要呈现插屏广告前，先通过isReady方法检查广告是否就绪
    if (_dmInterstitial.isReady)
    {
        [_dmInterstitial present]; // 呈现插屏广告
    }
    else
    {
        // 如果还没有ready，可以再调用loadAd
        [_dmInterstitial loadAd];
    }
}

// 当插屏广告被关闭后，回调该方法
- (void)dmInterstitialDidDismissScreen:(DMInterstitialAdController *)dmInterstitial
{
    // 插屏广告关闭后，加载一条新广告用于下次呈现
    [_dmInterstitial loadAd];
}
@end
```

关于 DMInterstitialAdController 的释放：
在这里强烈建议您，在释放 DMInterstitialAdController 之前，将其 delegate 属性设置为 nil。

设置可选参数：

与 DMAdView 相同，也可以为 DMInterstitialAdController 设置一些可选参数，来提高广告投放的精准性。这些参数的设置方法都定义在 DMInterstitialAdController.h 中，如下：

```
// 设置地理位置信息
- (void)setLocation:(CLLocation *)location;

// 设置邮编
- (void)setPostcode:(NSString *)postcode;

// 设置关键字
- (void)setKeywords:(NSString *)keywords;

// 设置用户年龄
- (void)setUserBirthday:(NSString *)userBirthday;

// 设置用户性别
- (void)setUserGender:(DMUserGenderType)userGender;
```

实现 DMInterstitialAdControllerDelegate：
您可以通过实现 DMInterstitialAdControllerDelegate 中定义的方法，来跟踪广告生命周期中的各个阶段。所有这些方法也都定义在 DMInterstitialAdController.h 中，如下：

```
@protocol DMInterstitialAdControllerDelegate <DMAdViewDelegate>
@optional
// 当插屏广告被成功加载后，回调该方法
- (void)dmInterstitialSuccessToLoadAd:(DMInterstitialAdController *)dmInterstitial;
// 当插屏广告加载失败后，回调该方法
- (void)dmInterstitialFailToLoadAd:(DMInterstitialAdController *)dmInterstitial withError:(NSError
*)err;

// 当插屏广告要被呈现出来前，回调该方法
- (void)dmInterstitialWillPresentScreen:(DMInterstitialAdController *)dmInterstitial;
// 当插屏广告被关闭后，回调该方法
- (void)dmInterstitialDidDismissScreen:(DMInterstitialAdController *)dmInterstitial;

// 当将要呈现出 Modal View 时，回调该方法。如打开内置浏览器。
- (void)dmInterstitialWillPresentModalView:(DMInterstitialAdController *)dmInterstitial;
// 当呈现的 Modal View 被关闭后，回调该方法。如内置浏览器被关闭。
- (void)dmInterstitialDidDismissModalView:(DMInterstitialAdController *)dmInterstitial;
// 当因用户的操作（如点击下载类广告，需要跳转到Store），需要离开当前应用时，回调该方法
- (void)dmInterstitialApplicationWillEnterBackground:(DMInterstitialAdController *)dmInterstitial;
@end
```

5.3 信息流广告的使用(具体使用请参考 Demo)

（信息流广告是针对 iphone/ipod 竖屏模式的下拉刷新定制）
信息流广告的使用，需要以下几步：

a. 导入 DMFeedsAdView.h 头文件。
b. 声明一个 DMFeedsAdView 的实例。
c. 让您的 UIViewController 实现 DMFeedsAdViewDelegate 协议。
d. 在您的 Controller 的 m 文件中，使用 PublisherId 和 PlacementId 以及负责呈现信息流广告的起始坐标 origin 来初始化一个 DMFeedsAdView，同时传递当前 controller,设置代理。
e. 在合适的时候，调用 loadAd 方法，来预加载信息流广告。
f. 在想要呈现信息流广告的时候，调用 present 方法，来呈现信息流广告。
g. 在呈现完成后，需要重新通过 loadAd 方法，加载一条新广告，用于下次呈现。
h. 注意：信息流只支持竖屏格式的展示，如果当前界面支持横屏 请做对应操作防止界面混乱。
    a) 广告在播放时，如果由竖屏转为横屏，请调用 closeAd 关闭广告

b) 如果在横屏状态下拉界面，请阻止广告展现

下面的代码片段，简单描述了这个过程：

```
#import <UIKit/UIKit.h>
#import "DMFeedsAdView.h"

@interface DMFeedsAdViewController : UIViewController<DMFeedsAdViewDelegate>
{

     DMFeedsAdView *_feedsView;

}
@property (nonatomic, retain) DMFeedsAdView *feedsView;

@end
```

```
   /**feeds ad view  **/
  _feedsView = [[DMFeedsAdView alloc]initWithPublisherId:DM_PUBLISHER_ID
                                                 placementId:DM_PLACEMENT_ID
                                                      origin:CGPointMake(0, 20)];

  _feedsView.rootViewController = self;
  _feedsView.delegate = self;

   [_feedsView loadAd];
```

我们建议对起始位置做 ios7 的适配，如果有 statusBar 不要忘了下移 20 像素高度。

```
- (void)reloadTableViewDataSource{

    //  should be calling your tableviews data source model to reload
    //  put here just for demo
    _reloading = YES;

    /** 竖屏情况开始展现广告
     present feeds ad view
     please choose the orientation you support
     UIDeviceOrientationPortrait/UIDeviceOrientationPortraitUpsideDown   **/

    if ([[UIApplication sharedApplication] statusBarOrientation]
                                          == UIInterfaceOrientationPortrait)
    {
        [_feedsView present];
    }


}
```

竖屏状态用户下拉的时候调用 present 方法（我们只支持竖屏，所以请选择您所支持的竖屏模式进行判定后进行相应的 present 操作）。

鉴于显示广告时程序界面依然可操作，所以我们提供了关闭 Feeds ad view 的方法 closeAd。需要开发者在出现视图切换时手动关闭广告。（例：用户点击 cell 进入下一个界面,用户从竖屏转向横屏操作）

```
- (void)didRotateFromInterfaceOrientation:(UIInterfaceOrientation)fromInterfaceOrientation
{
    //close the ad view by yourself
    [_feedsView closeAd];
}
```

同时我们建议在视图消失时实现加载下一条广告

```
- (void)dmFeedsDidDismissScreen:(DMFeedsAdView *)dmFeeds
{
    [UIView beginAnimations:nil context:NULL];
    [UIView setAnimationDuration:.3];
    [self.myTable setFrame:CGRectMake(origin.x,
                                      origin.y,
                                      self.myTable.frame.size.width,
                                      self.myTable.frame.size.height)];
    [UIView commitAnimations];
    //load ad when the ad view dismiss
    [_feedsView loadAd];
}
```

关于 DMFeedsAdView 的释放：
在这里强烈建议您，在释放 DMFeedsAdView 之前，将其 delegate 属性设置为 nil。
设置可选参数：
与 DMAdView 相同，也可以为 DMFeedsAdView 设置一些可选参数，来提高广告投放的精准性。这些参数的设置方法都定义在 DMFeedsAdView.h 中，
如下：

```
// 设置地理位置信息
- (void)setLocation:(CLLocation *)location;

// 设置邮编
- (void)setPostcode:(NSString *)postcode;

// 设置关键字
- (void)setKeywords:(NSString *)keywords;

// 设置用户年龄
- (void)setUserBirthday:(NSString *)userBirthday;

// 设置用户性别
- (void)setUserGender:(DMUserGenderType)userGender;
```

实现 DMFeedsAdViewDelegate：
您可以通过实现 DMFeedsAdViewDelegate 中定义的方法，来跟踪广告生命周期中的各个阶段。所有这些方法也都定义在 DMFeedsAdView.h 中，如下：

```objc
@protocol DMFeedsAdViewDelegate

@optional
// Sent when an ad request success to load an ad
- (void)dmFeedsSuccessToLoadAd:(DMFeedsAdView *)dmFeeds;
// Sent when an ad request fail to load an ad
- (void)dmFeedsFailToLoadAd:(DMFeedsAdView *)dmFeeds withError:(NSError *)err;
// Sent when the feeds ad is clicked
- (void)dmFeedsDidClicked:(DMFeedsAdView *)dmFeeds;

// Sent just before presenting the user a modal view
- (void)dmFeedsWillPresentModalView:(DMFeedsAdView *)dmFeeds;
// Sent just after dismissing the modal view
- (void)dmFeedsDidDismissModalView:(DMFeedsAdView *)dmFeeds;
// Sent just before the application will background or terminate because the user's action
// (Such as the user clicked on an ad that will launch App Store).
- (void)dmFeedsApplicationWillEnterBackground:(DMFeedsAdView *)dmFeeds;

// Sent just before presenting an feeds ad view
- (void)dmFeedsWillPresentScreen:(DMFeedsAdView *)dmFeeds;
// Sent just after dismissing a feeds ad view
- (void)dmFeedsDidDismissScreen:(DMFeedsAdView *)dmFeeds;

@end
```

5.4 开屏广告的使用

开屏广告的使用场景是在应用刚刚开启时，在 SDK 内分为缓存开屏和实时开屏两种。缓存开屏的加载机制为，本次加载，缓存本次的广告应答以及相关资源，下次开启时展现。而实时开屏在每次开启时都会实时请求广告并加载和缓存资源，在超时时间前加载完成即自动展现。详细使用方法可见 Sample 中的 App 委派类的 `application:didFinishLaunchingWithOptions:` 方法中使用的示例代码。

	广告请求流程	优点	缺点
缓存开屏	当次缓存，下次展现	无需等待广告loading,体验好。	填充→展现流失率较高
实时开屏	当次加载，当次展现，超时放弃（超时时限开发者可设）	填充→展现流失率较低。	用户需要等待

如果您想让开屏广告与您的启动图片融合的更好，获得更好的用户体验，请参考 Sample 项目中的做法。

设置可选参数：
与其他广告形式相同，也可以为 DMSplashAdController 设置一些可选参数，来提高广告投放的精准性。这些参数的设置方法都定义在 DMSplashAdController.h 中，如下：

```objc
// 设置地理位置信息
- (void)setLocation:(CLLocation *)location;

// 设置邮编
- (void)setPostcode:(NSString *)postcode;

// 设置关键字
- (void)setKeywords:(NSString *)keywords;

// 设置用户年龄
- (void)setUserBirthday:(NSString *)userBirthday;
```

```
// 设置用户性别
- (void)setUserGender:(DMUserGenderType)userGender;
```

实现 DMSplashAdControllerDelegate：
您可以通过实现 DMSplashAdControllerDelegate 中定义的方法，来跟踪广告生命周期中的各个阶段。所有这些方法也都定义在 DMSplashAdController.h 中，如下：

```
@protocol DMSplashAdControllerDelegate
@optional
// 当开屏广告加载成功后，回调该方法
- (void)dmSplashAdSuccessToLoadAd:(DMSplashAdController *)dmSplashAd;
// 当开屏广告加载失败后，回调该方法
- (void)dmSplashAdFailToLoadAd:(DMSplashAdController *)dmSplashAd withError:(NSError *)err;

// 当插屏广告要被呈现出来前，回调该方法
- (void)dmSplashAdWillPresentScreen:(DMSplashAdController *)dmSplashAd;
// 当插屏广告被关闭后，回调该方法
- (void)dmSplashAdDidDismissScreen:(DMSplashAdController *)dmSplashAd;
@end
```

6 开发者工具的使用

6.1 评价提醒

只需简单几步，即可实现好评提醒功能：

```
// 检查更新提醒，此处使用的是测试ID，请登陆多盟官网（www.domob.cn）获取新的ID
DMTools *_dmTools = [[DMTools alloc] initWithPublisherId:@"56OJyM1ouMGoULfJaL"];
[_dmTools checkRateInfo];
[_dmTools release];
```

若需更多帮助，请登陆多盟官网（http://www.domob.cn/help/index.htm）。

7 其他

更详细的使用方法，请参考发行文件中的 Sample 程序。
如还有疑问，欢迎随时发邮件到 support@domob.cn 获得更多帮助。