NATACORP
CRAZY TECHNOLOGY FOR FUN LIFE

AdsNaTa iOS SDK Integration Guide 4.1.6

The following document provides detailed instructions on how to integrate the AdsNaTa iOS SDK 4.1.6 into your iOS 4.3 to iOS 6.1.3 Projects based on a simple Demo app. The AdsNaTa SDK is capable of displaying both traditional Banner Ads and (Rich Media) Interstitials.

Step 1: Set Up your Application in your AdsNaTa Control Panel

- **Log into your AdsNaTa Administration Control Panel**
- **Click on Inventory>Add Publications if you'd like to create a new Publication**
- **If you want to integrate an existing Publication, navigate to Inventory > Integration**
- **You will be provided with a unique Placement ID (called Publisher ID in code) and a Request URL**

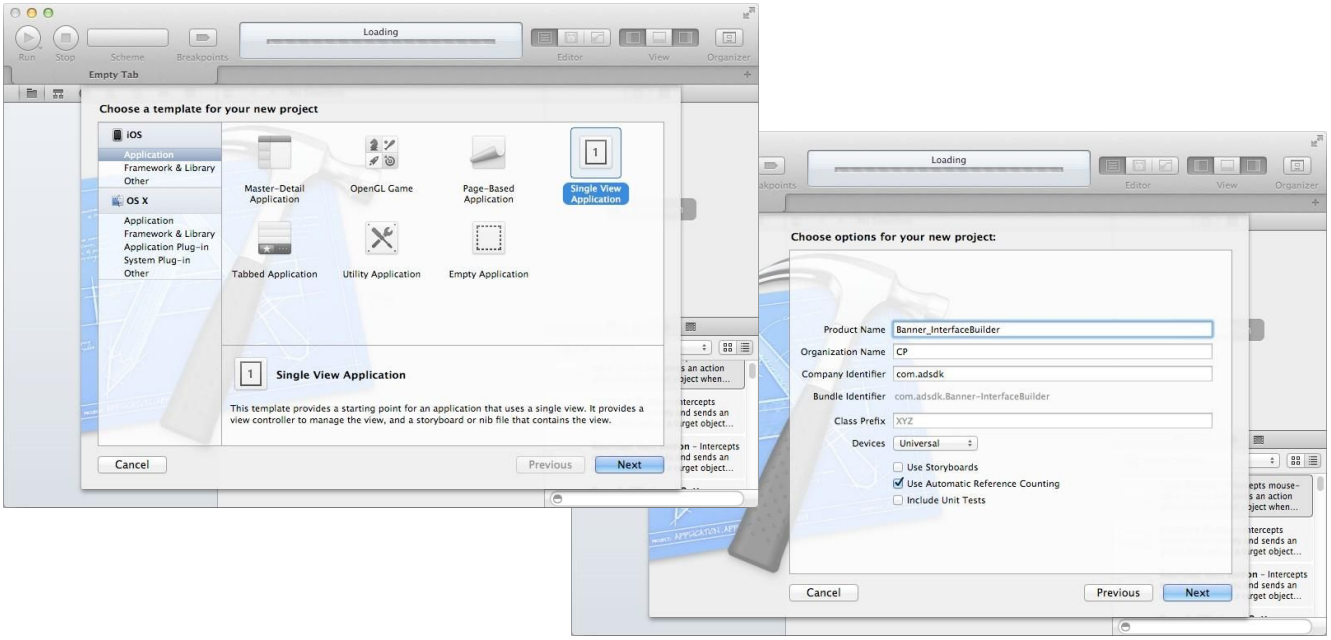
Step 2: Download the SDK

The downloaded ZIP contains the following files:

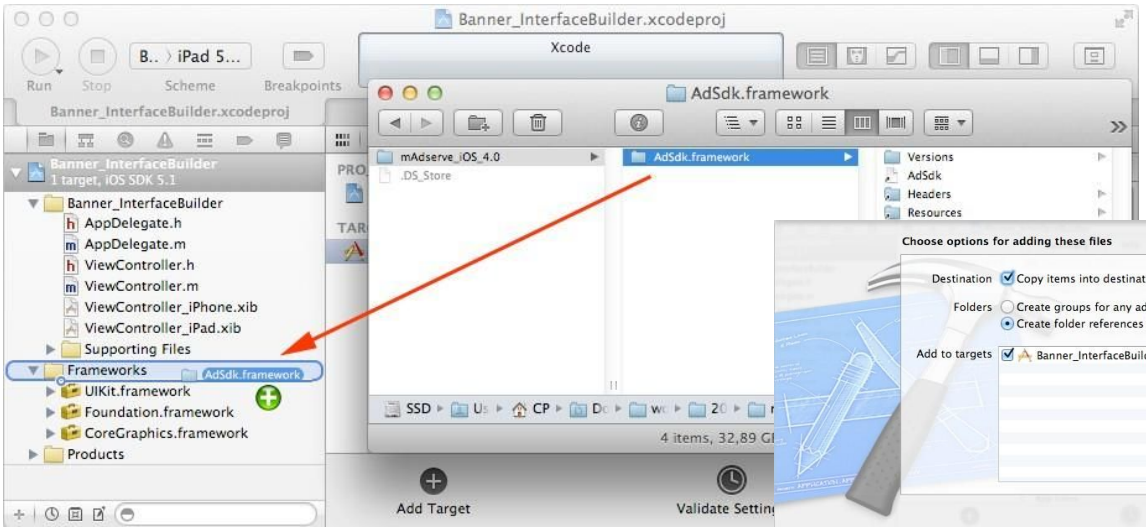
- **PDF Documentation "AdsNaTa iOS SDK Setup Guide 4.1.6.pdf" (this document)**
- **The SDK Framework itself: AdSdk.framework**
- **Demo Application "AdSdkDemo" folder:**
  - **Request Banner Ads and Interstitial Ads with Buttons**
  - **Based on Xcode 4.6.1**
- **"Source_Code" folder with with 3 additional here described Coding and Customization options**

Step 3: Create and define a new iOS Project in Xcode

Create a New Project "Banner_InterfaceBuilder", select "Single View Application", use no "Storyboard" and use "Automatic Reference Counting" (find the complete Source Code at the "Source_Code" folder).

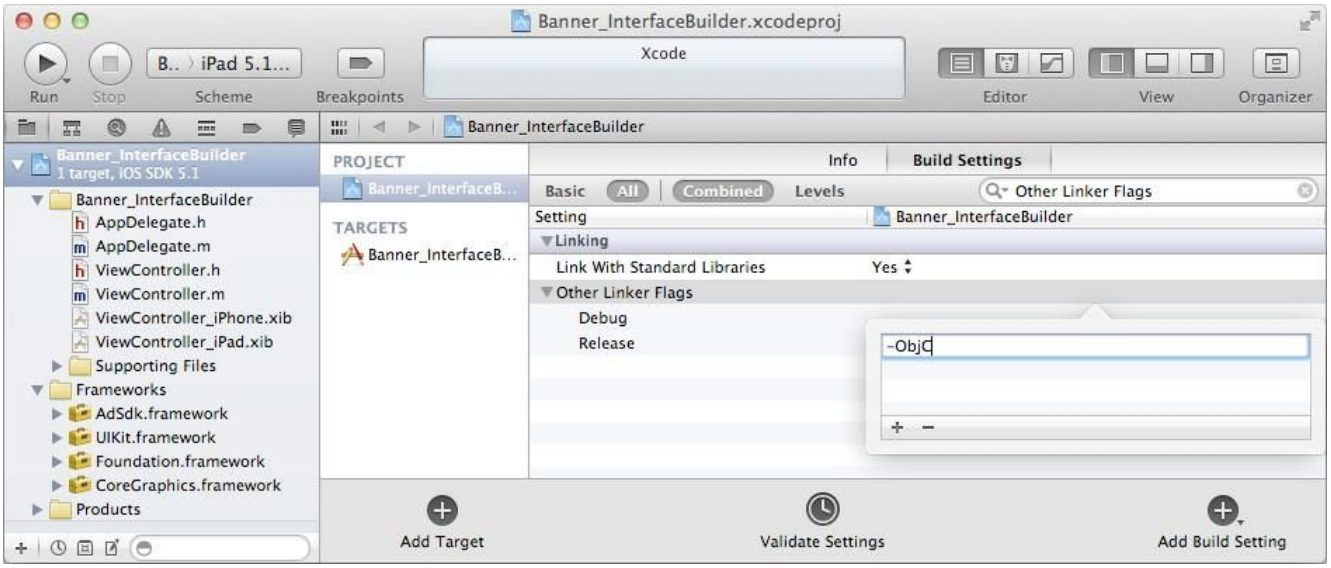


Step 4: Add the SDK to your project "Frameworks" folder

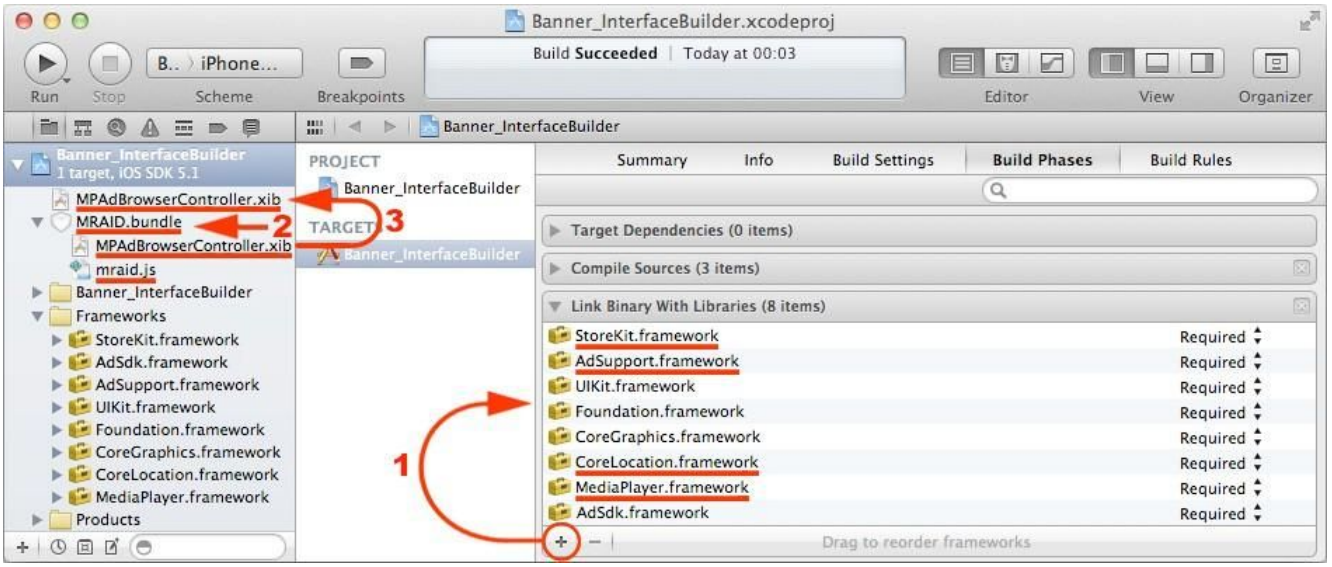


Drag & Drop the AdSdk Framework from the Finder into the Frameworks Folder in your Xcode project

Under "Project" > "Build Settings" search for"Other Linker Flags" and add "-ObjC" to load objective-C code from libraries. Note: For non ARC Projects add an additional line "-fobjc-arc".



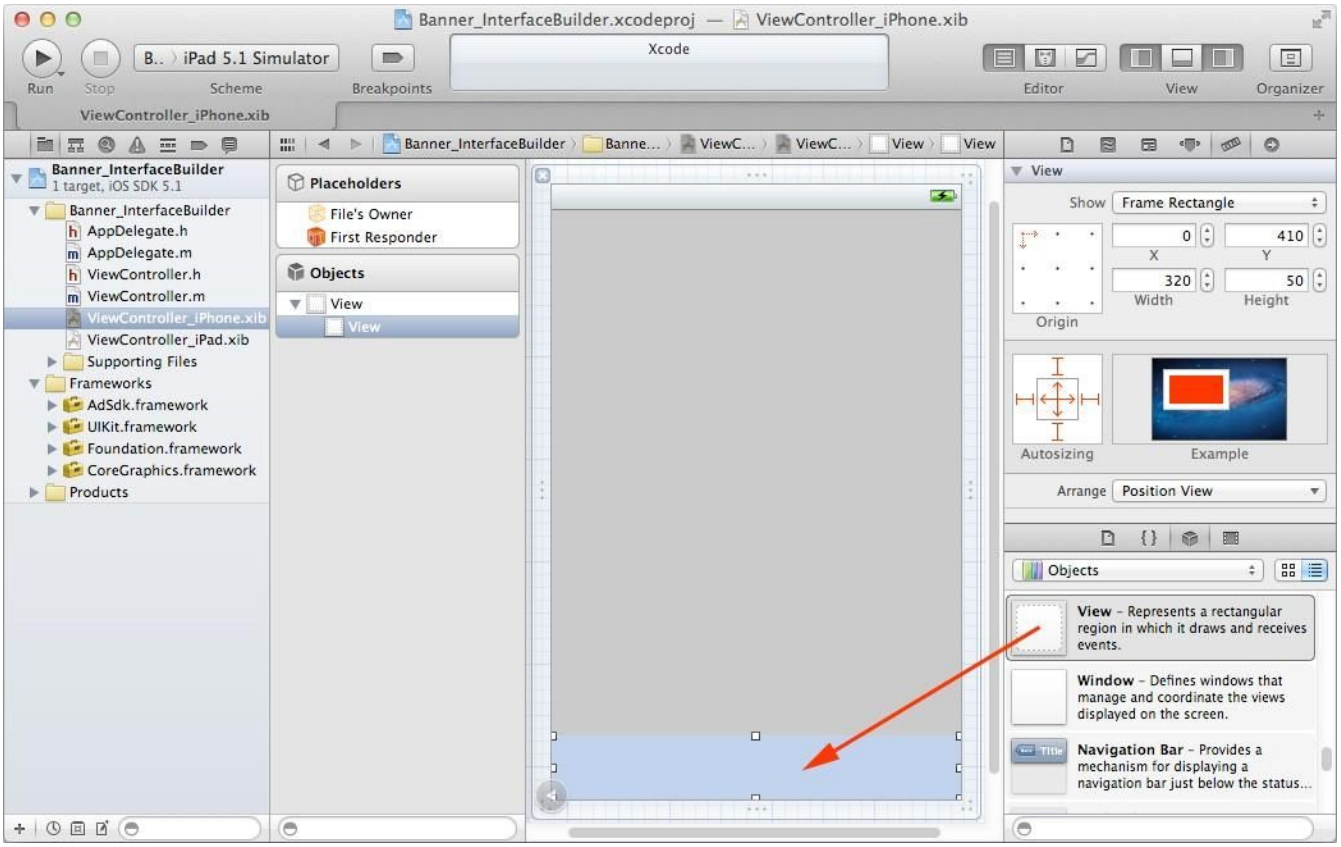
1) Add mandatory Frameworks, 2) add MRAID.bundle and 3) copy MPAdBrowserController.xib into root.



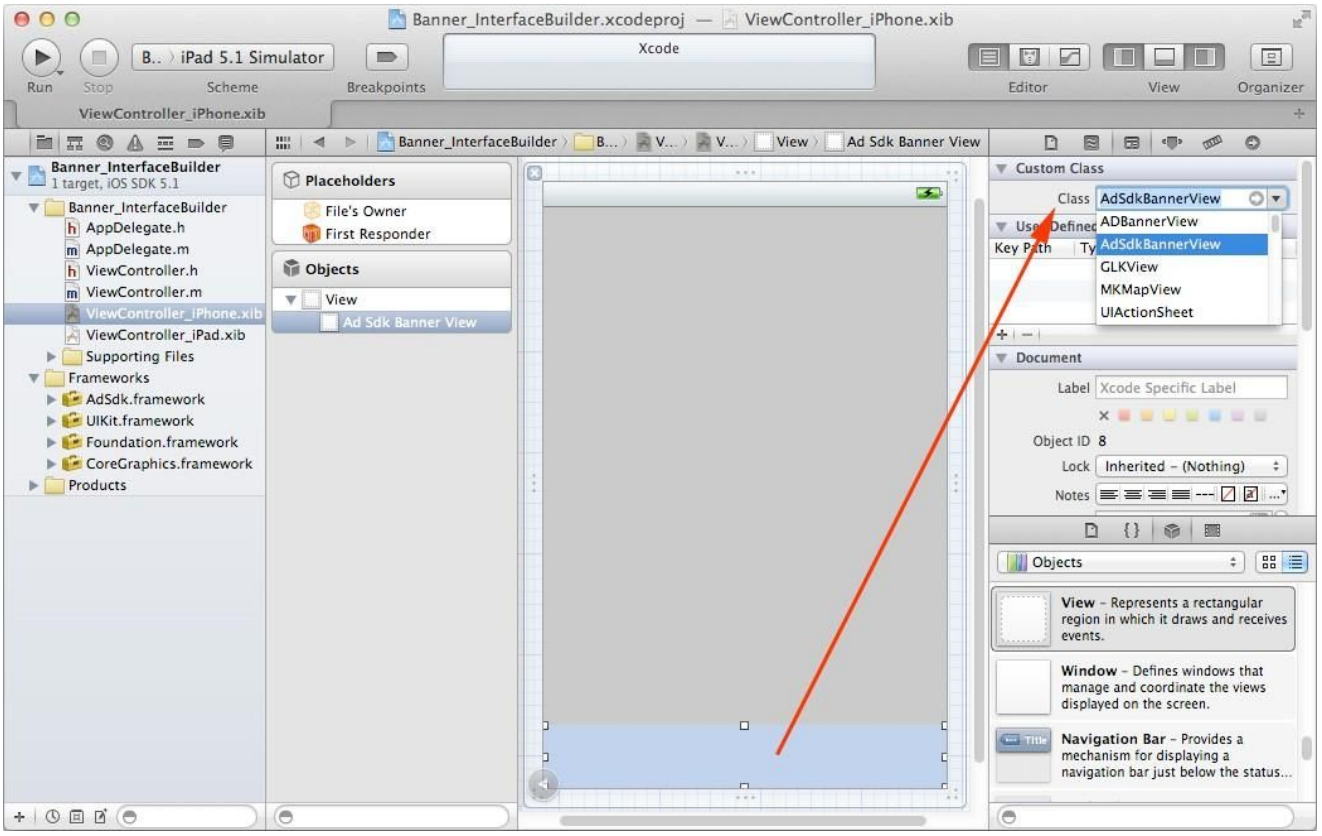
To display Banner Ads please continue with "Step 5". To display Interstitial Ads please jump to "Step 6".

Step 5: Request and display Banner Ads using Interface Builder

Go to the“ViewController_iPhone.xib” and add a View sized 320*50 Go to the“ViewController_iPad.xib” and add a View sized 728*90

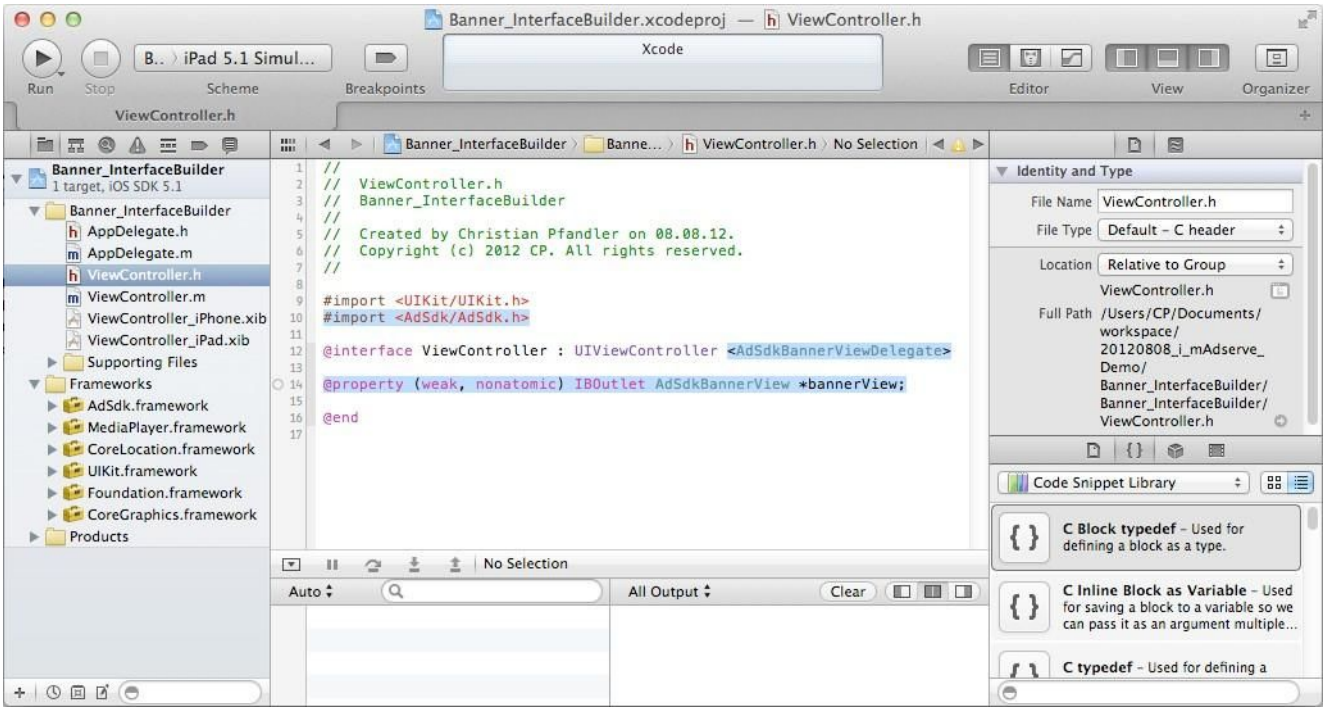


Set the Class of this View to“AdSdkBannerView”. Do the same for the“ViewController_iPad.xib”.

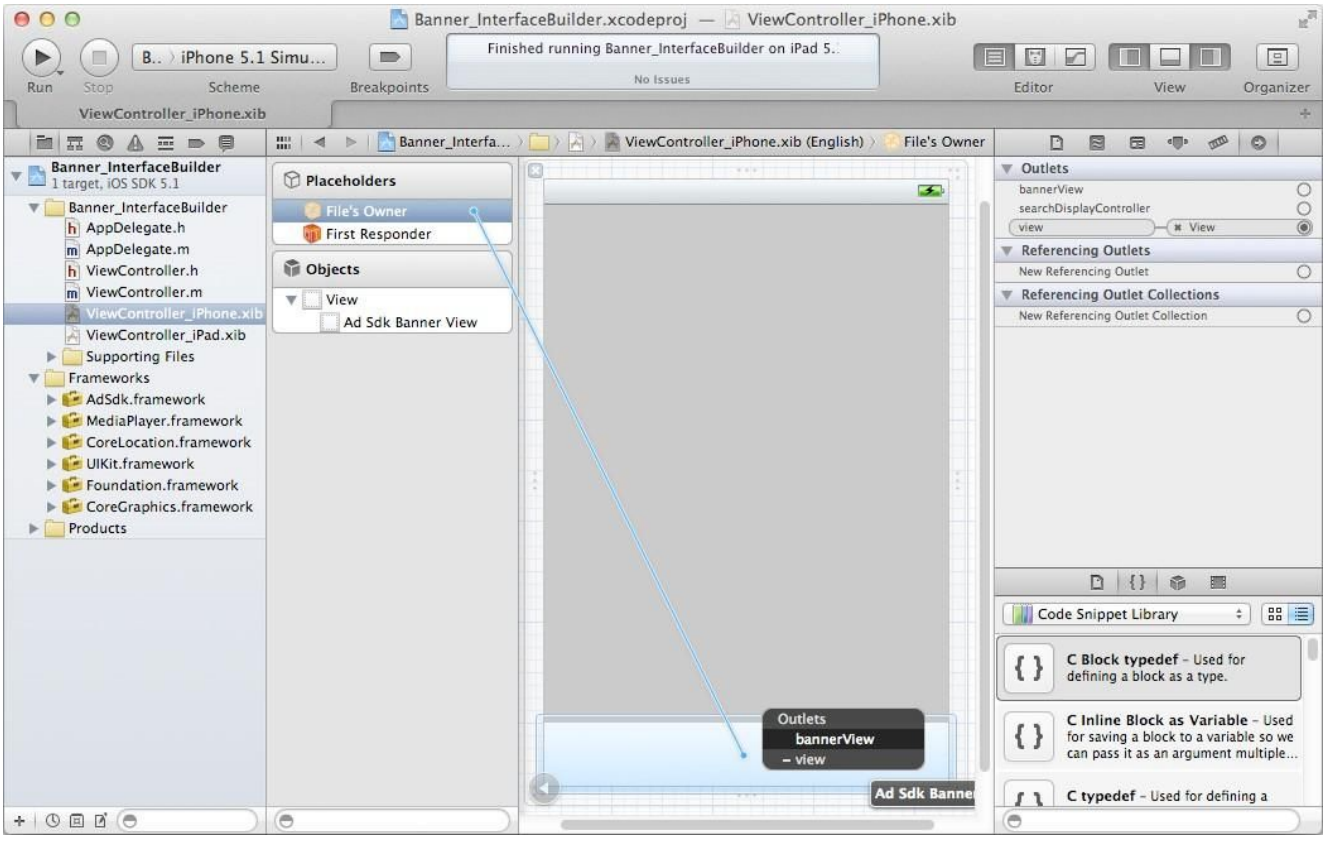


Jump to ViewController.h and add the lines (copy/paste from "Source_Code" folder).
Note the property for the bannerView to reference it from code. Don't forget to `@synthesize` in .m file.

```
#import <UIKit/UIKit.h>
#import <AdSdk/AdSdk.h>
        @interface ViewController : UIViewController <AdSdkBannerViewDelegate>
        @property (weak, nonatomic) IBOutlet AdSdkBannerView *bannerView;
@end
```



In the"ViewController_iPhone.xib" assign by ctrl-drag the "File's Owner" to the BannerView and select "bannerView". Do the same for the"ViewController_iPad.xib".



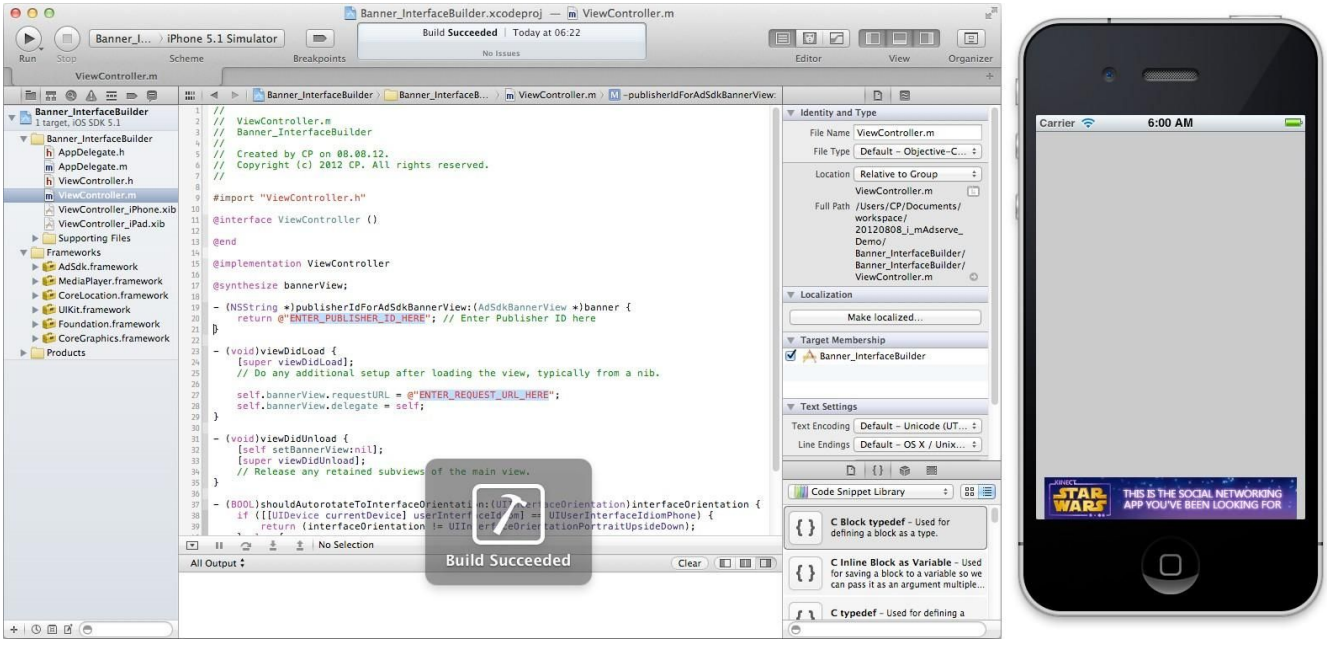
In ViewController.m add the following lines (copy/paste from "Source_Code" folder). Note: The Placement ID is called Publisher ID in code!

```
-       (NSString *)publisherIdForAdSdkBannerView:(AdSdkBannerView
        *)banner { return @"ENTER_PUBLISHER_ID_HERE"; // Enter
        Publisher ID here
}
```

The final step is to set the RequestURL of your AdsNaTa ad server and set the bannerView delegate to self. This will initiate an ad request.

```
- (void)viewDidLoad {
        [super viewDidLoad];
        // Do any additional setup after loading the view, typically from a nib.

        self.bannerView.requestURL =
        @"ENTER_REQUEST_URL_HERE"; self.bannerView.delegate
        = self;
}
```

Build with your Placement ID (Publisher ID in code), compare with provided code in "Source_Code" folder.



Please find in the "Source_Code" folder two additional Demos including Banner Ads:
AdSdkDemo_Banner

- **The quickest and simplest way to add a Banner Ad in your app by just using code**
- **Shows several customizations over the Banner_InterfaceBuilder version:**
    - **The banner view is created and placed below the bottom of the view**
    - **This allows to animate it into place once an ad is successfully retrieved**
    - **The refresh animation property is set to `UIViewAnimationTransitionCurlDown`.**
    - **The background Color is set to dark gray to match the UI**

AdSdkDemo

- **The best way to test your Placement ID (Publisher ID in code) to request Banner or Interstitial Ads**
- **A more advanced approach which allows the Banner to be added and removed dynamically**
- **Use this approach when integrating both Banner and Interstitial Ads into the same view**
- **Use where it is important to hide banner adverts while showing interstitial Ads**

Note: Instead of requesting an Ad using a button use the code under the `- (IBAction)requestBannerAdvert`

If you don't want to add Interstitial Ads please jump to "Step 7".

Step 6: Request and display Interstitial Full-Screen Ads

Create a New Project called "AdSdkDemo_VideoInterstitial", follow Step 3, Step 4 and these steps touched on here to reach the provided state you find in the provided "Source_Code" folder.

Ad these lines to ViewController.h (copy/paste from "Source_Code" folder). Note the property for the videoInterstitialViewController to reference it from code. Don't forget to `@synthesize` in .m file.

```
#import <UIKit/UIKit.h>
#import <AdSdk/AdSdk.h>
@interface ViewController : UIViewController <AdSdkVideoInterstitialViewControllerDelegate>
@property (nonatomic, strong) AdSdkVideoInterstitialViewController
       *videoInterstitialViewController
                        ;
-   (IBAction)requestInterstitialAdvert:(id)sender;
@end
```

Ad these lines to ViewController.m to create the Interstitial Ad view Controller and add the view(copy/paste from "Source_Code" folder).

```
-   (void)viewDidLoad
   { [super
   viewDidLoad];
   // Create, add Interstitial Ad View Controller and add view to view hierarchy
   self.videoInterstitialViewController = [[AdSdkVideoInterstitialViewController alloc]
   init];
   // Assign delegate
   self.videoInterstitialViewController.delegate =
   self;
   // Defaults to NO. Set to YES to get locationAware Adverts
   self.videoInterstitialViewController.locationAwareAdverts =
   YES;
   // Add view. Note when it is created is transparent, with alpha = 0.0 and hidden
   // Only when an ad is being presented it become visible
   [self.view
   addSubview:self.videoInterstitialViewController.view];
}
```

Continue to add an IBAction to the ViewController.h file (copy/paste from "Source_Code" folder)

```
- (IBAction)requestInterstitialAdvert:(id)sender
   { if(self.videoInterstitialViewController) {
      self.videoInterstitialViewController.requestURL =
      @"ENTER_REQUEST_URL_HER"; [self.videoInterstitialViewController
      requestAd];
   }
}
```

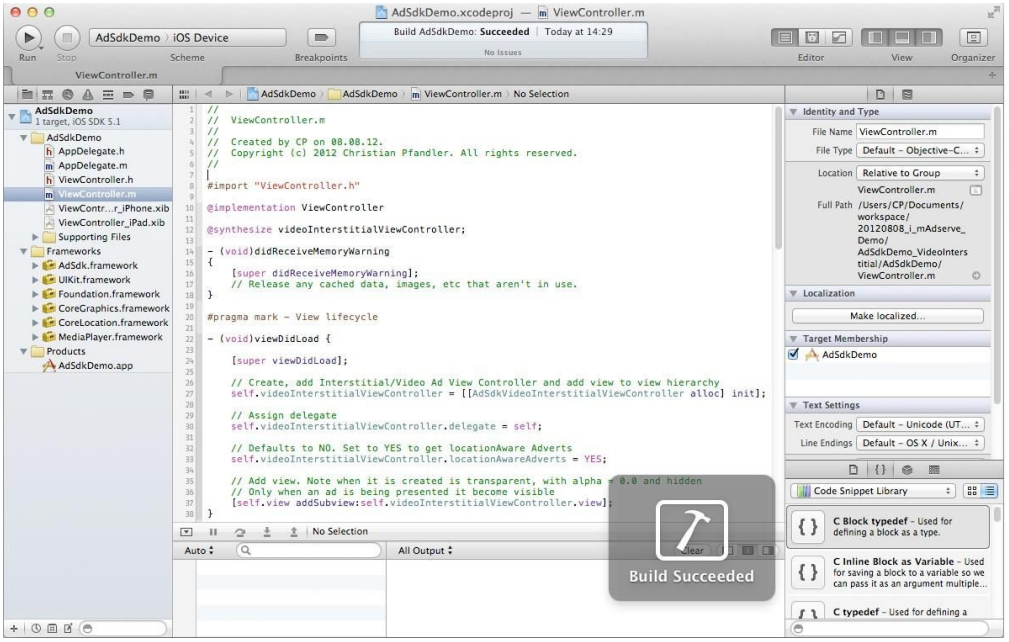
Jump to ViewController.xib and add a Button attached to the IBAction. Instead of requesting an Ad using a button use the code under the `- (IBAction)requestInterstitialAdvert`

Continue to modify ViewController.m by adding the following Interstitial delegate methods (copy/paste from "Source_Code" folder). Placement ID is called Publisher ID in code.

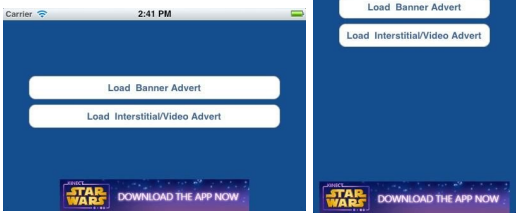
```
#pragma mark AdSdk Interstitial Delegate Methods
-       (NSString *)publisherIdForAdSdkVideoInterstitialView:
        (AdSdkVideoInterstitialViewController *)videoInterstitial
        {
   return @"ENTER_PUBLISHER_ID_HERE"; // Enter Publisher ID here
```

```
}
- (void)adsdkVideoInterstitialViewDidLoadAdSdkAd:(AdSdkVideoInterstitialViewController *)
      videoInterstitial advertTypeLoaded:(AdSdkAdType)advertType

   { NSLog(@"AdSdk Interstitial: did load ad");

   // Means an advert has been retrieved and configured.
   // Display the ad using the presentAd method and ensure you pass back the advertType

   [videoInterstitial presentAd:advertType];
}

- (void)adsdkVideoInterstitialView:(AdSdkVideoInterstitialViewController
      *) banner didFailToReceiveAdWithError:(NSError *)error {

   NSLog(@"AdSdk Interstitial: did fail to load ad: %@", [error localizedDescription]);
}
```

Build with your Placement ID (Publisher ID in code), compare with provided code in "Source_Code" folder.



Step 7: Finished!!!



If you already created a campaign in your AdsNaTa Dashboard, you should now see the ad.

Note: Please always clear memory before showing Ads and restore your apps orientation after an Ad is shown.

iPhone 5, iOS 6 and Xcode 4.5 compatibility and new Properties

AdsNaTa SDK now works with the new iPhone 5 in "letterbox" and takes full advantage of the new 4" display (if you include a Default-568@2x.png launch image).

AdsNaTa SDK 4.1.6 has been compiled with Xcode 4.5 and includes armv7 and armv7s (iPhone 5) versions for this devices and above: iPhone 3GS, iPad 2 and iPod Touch (Sept 2010). Minimum Deployment Target is now iOS 4.3. Please note that Xcode 4.5 no longer includes a iOS 4.3 Simulator so testing should be done on an actual iOS 4.3 device.

iOS 6.X devices the SDK also uses the new identifierForAdvertising instead of OpenUDID or MAC hashing.

BannerView - allowDelegateAssigmentToRequestAd:

Previous assigning the delegate caused a requestAd. Now gain finer control by setting the property allowDelegateAssigmentToRequestAd to NO, assign the delegate and use the requestAd method to load an advert when ready. Use the previous by setting it to YES.

BannerView - adsdkBannerViewDidLoadRefreshedAd:

This is called each time when a Banner Ad is refreshed

BannerView - requestAd:

This is used to manually request an Ad

- **Please hide Ad banner if no ad is retrieved (displaying an empty area is against Apple guidelines)**
- **Ads may be smaller than the actual view. Set banner background color to match your UI or"clearColor"**
- **To implement both Banner & vAds refer to the provided AdSdkDemo Source Code**

Explanation of Methods for Banner Ads

Mandatory! You must implement and set the Placement ID (Publisher ID in code)

```
- (NSString *)publisherIdForAdSdkBannerView:(AdSdkBannerView *)banner;
```

Optional. Please use this methods for a smoother integration and result

```
// Called if an Ad has been successfully retrieved and displayed the first time.
// Not called when an adView receive a "refreshed" Ad
-       (void)adsdkBannerViewDidLoadAdSdkAd:(AdSdkBannerView *)banner;

// Called if an existing Ad view receives a "refreshed" Ad
- (void)adsdkBannerViewDidLoadRefreshedAd:(AdSdkBannerView *)banner;

// Called if no banner is available or there is an error
-       (void)adsdkBannerView:(AdSdkBannerView
        *)banner didFailToReceiveAdWithError:(NSError
        *)error;

// Called when user taps on a banner
-       (BOOL)adsdkBannerViewActionShouldBegin:(AdSdkBannerView
        *)banner willLeaveApplication:(BOOL)willLeave;

// Called when the modal web view will be displayed
- (void)adsdkBannerViewActionWillPresent:(AdSdkBannerView *)banner;

// Called when the modal web view is about to be cancelled
// Restart any foreground activities paused as part of adsdkBannerViewActionWillPresent:
- (void)adsdkBannerViewActionWillFinish:(AdSdkBannerView *)banner;

// Called when the modal web view is cancelled and the user is returning to the app
- (void)adsdkBannerViewActionDidFinish:(AdSdkBannerView *)banner;

// Called when a user tap results in Application Switching
- (void)adsdkBannerViewActionWillLeaveApplication:(AdSdkBannerView *)banner;
```

Explanation of Methods for Interstitial Ads

Mandatory! You must implement and set the Placement ID (Publisher ID in code)

```
-       (NSString *) publisherIdForAdSdkVideoInterstitialView:
        (AdSdkVideoInterstitialViewController *) videoInterstitial;
```

Optional. Please use this methods for a smoother integration and result

```
// Called if a Interstitial Ad has been successfully retrieved and
// is ready to display via - (void)presentAd:(AdSdkAdType)advertType:
-       (void)adsdkVideoInterstitialViewDidLoadAdSdkAd:(AdSdkVideoInterstitialViewControl
        ler *) videoInterstitial advertTypeLoaded:(AdSdkAdType)advertType;

// Called if no Interstitial is available or there is an error
-       (void)adsdkVideoInterstitialView:(AdSdkVideoInterstitialViewControl
        ler *) banner didFailToReceiveAdWithError:(NSError *)error;

// Sent immediately before Interstitial is shown to the user. At this point
// pause any animations, timers or other activities that assume user interaction and
// save app state, much like on UIApplicationDidEnterBackgroundNotification.
// Remember that the user may press Home or touch links to other apps like App Store or
// iTunes within the interstitial, leaving your app.
-       (void)adsdkadsdkVideoInterstitialViewActionWillPresentScre
        en: (AdSdkVideoInterstitialViewController
        *)videoInterstitial;

// Sent immediately before interstitial leaves the screen. At this point
// restart any foreground activities paused as part of interstitialWillPresentScreen:
-       (void)adsdkVideoInterstitialViewWillDismissScreen:(AdSdkVideoInterstitialViewContro
        ller *) videoInterstitial;
```

```
// Sent when the user has dismissed interstitial and it has left the screen.
-       (void)adsdkVideoInterstitialViewDidDismissScreen:(AdSdkVideoInterstitialViewControl
        ler *) videoInterstitial;

// Called when a user tap results in Application Switching
-       (void)adsdkVideoInterstitialViewActionWillLeaveApplication
        : (AdSdkVideoInterstitialViewController
        *)videoInterstitial;
```