

Avazu Native Ad SDK Setup for iOS

Date	Version	Description	Developer
2015.03.12	v1.0.0	First Version	曹诗聪
2015.03.25	v1.0.1	Optimization and issue fix	曹诗聪
2015.03.27	v1.0.2	Sample Project Optimization	曹诗聪
2015.05.22	v1.1.0	Change Server	曹诗聪

1. Introduction	3
1.1 What the SDK can provide	3
1.2 Compatibility	3
1.3 Development environment.....	3
2. Configuration	4
2.1 Traffic Source ID	4
2.2 Adding the SDK to your project.....	4
2.3 Adding Frameworks	4
2.3 Adding Linker Flag	5
3. Show AD	5
3.1 Creating AvazuADView by code	5
3.1.1 Creating a instance for AvazuADView	5
3.1.2 Choosing a supported ADtype	6
3.1.3 Setting customized Advview	7
3.2 Creating AvazuADView by Interface Builder	8
4. Implementing the delegate	9

1.Introduction

1.1 What the SDK can provide

Avazu ADSDK can provide following functions in your mobile iOS App:

- Show Single Banner (Recommended size: 320 * 100)
- Show Transparent Banner (Recommended size: 410* 100)
- Show Banner App Wall
- Show Single Line Button APP Wall
- Show Multiple Line Button APP Wall

For the ad types, the appearance and size of the ad are fully customizable with our function provided in the SDK.

1.2 Compatibility

The minimum runtime OS requirement is iOS 6.0, or higher, iOS 8.x is fully supported.

1.3 Development environment

Operation System: macosx lion, or higher

Development Tool: Xcode 5.0, or higher

2. Configuration

2.1 Traffic Source ID

Make sure you are have registered Avazu APX account and have a valid Traffic Source ID for display.

2.2 Adding the SDK to your project

- [libAvazuAdSDK.a](#)
- [AvazuADView.h](#)

2.3 Adding Frameworks






For the SDK to work correctly , adding some frameworks to your Xcode project is required.

The frameworks required to compile the AvazuADSDK library are given below:

- [CoreTelephony.framework](#)
- [UIKit.framework](#)
- [Foundation. framework](#)
- [AdSupport. framework](#)

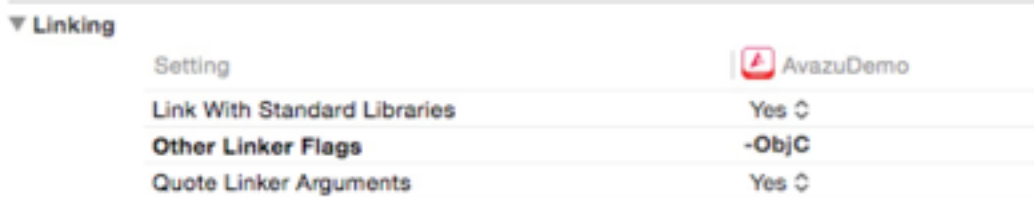
▼ Link Binary With Libraries (5 items)



Name	Status
 CoreTelephony.framework	Required ⇅
 libAvazuAdSDK.a	Required ⇅
 UIKit.framework	Required ⇅
 Foundation.framework	Required ⇅
 AdSupport.framework	Required ⇅
+ — Drag to reorder frameworks	

2.3 Adding Linker Flag

Add the -ObjC flag to the Other Linker Flags.



3. Show AD

[AvazuADSDK](#) provide 5 different types of ads, which include single banner, banner app wall, transparent banner, single line button App wall and multiple-line button App wall.

As a developer, you can simply create these advert type with one interface: [AvazuADView](#), it is simply a [UIView](#) subclass displaying HTML5 ads that respond to user touch. It is easy to create [AvazuADView](#) in code.

To get a valid ad, please perform one of following two methods:

3.1 Creating AvazuADView by code

3.1.1 Creating a instance for AvazuADView

Assuming you already had a appropriate [Traffic Source ID](#), and you will use it to create the instance every time, so make sure the [Traffic Source ID](#) is valid, or AvazuADSDK will fail to load the ads.

Follow the below steps:

1. Import the [AvazuADView.h](#) file.
2. Declare a instance in the header file. Your header file should look like this:

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

@interface ViewController : UIViewController <AvazuADViewDelegate>
@property (strong, nonatomic) AvazuADView *adView;
@end
```

3. Create an [AvazuADView](#) instance using the following method:

```
self.adView = [[AvazuADView alloc] initWithFrame:adFrame
               adType:AVAZU_SINGLE_BANNER
               sourceID:@"input your Traffic Source ID here"];
```

4. After creating an `adView` instance, set the delegate property of `AvazuADView` as your view controller before loading the ad. This is so that you will be notified about success or failure.
5. Add your `adView` to the main view before loading the ad.
6. Setup the `adView` according to your requirement, we will discuss this in detail in chapter 3.1.2.
7. load the ad using this method: `[self.adView loadAD]`

Your `viewController` class should look like below:

```
- (void)viewDidLoad {
    [super viewDidLoad];
    CGRect adframe = CGRectMake(0.0, 0.0, 320.0, 100.0);
    self.adView = [[AvazuADView alloc] initWithFrame:adframe
               adType:AVAZU_SINGLE_BANNER
               sourceID:@"1234"];

    self.adView.delegate = self;
    [self.view addSubview:self.adView];
    //Insert Customized Adview setup here
    [adView loadAD];
}
```

8. We recommend that you set the delegate to nil in the `dealloc` method of your `ViewController`, or at any time when you are releasing the `adView`.

```
- (void)dealloc {
    self.adView.delegate = nil;
}
```

3.1.2 Choosing a supported ADtype

Select the ADtype value from the following set of values declared in the `AvazuADView` method:

AD Type	Value	Variable Name
Single Banner	9	AVAZU_SINGLE_BANNER
Banner App Wall	10	AVAZU_BANNER_APPWALL
Transparent Banner	11	AVAZU_TRANSPARENT_SINGLE_BANNER
Single Line Button App Wall	12	AVAZU_SINGLE_BUTTON_APP_WALL
Multiple Line Button App Wall	13	AVAZU_MULTIPLE_BUTTON_APP_WALL

3.1.3 Setting customized Advview

AvazuADSDK support fully customization for different AD types, to apply the customized settings, you just need one line code like:

```
self.adView.isNeedIcon = 0;
```

1. Setting customized Ad Elements

Value Name	Value Type	Default Value	Usage
isNeedIcon	BOOL	1	Show app's icon or not in advview
isNeedTitle	BOOL	1	Show app's title or not in advview
isNeedCat	BOOL	1	Show app's category or not in advview
isNeedSize	BOOL	1	Show app's size or not in advview
isNeedRating	BOOL	1	Show app's rating or not in advview
isNeedInstallButton	BOOL	1	Show install button or not in advview
isNeedReviewNumber	BOOL	1	Show app's review number or not in advview
isNeedLoadingIndicator	BOOL	1	Show loading indicator or not when loading ad

2. Setting customized Ad Color

Value Name	Value Type	Default Value	Usage
blockBackColor	NSString	null	Set the background color of the ad block
appTitleColor	NSString	null	Set the font color of the app title
buttonBackColor	NSString	null	Set the font color of download button
buttonTextColor	NSString	null	Set the font color of download button
mainBackColor	NSString	null	Set the background color of app wall

Note: When setting AD color, All color value must be a NSString **started with “#” followed by 6 hexadecimal RGB digitals**. For example, #FFFFFF represents for white color while #000000 represents for color black.

3. Setting App count

Value Name	Value Type	Default Value	Usage
appCount	INT	1	Set the amount of app in advview

Note: appCount is required for these ad types:

Banner App Wall, Single Line Button App Wall and Multiple Line Button App Wall.

4. Setting Transparency Ratio For Transparent Banner

Value Name	Value Type	Default Value	Usage
transparentBannerAlpha	INT	0	Set the transparency ratio of the background for transparent banner

Note: transparentBannerAlpha is **required only for Transparent Banner**, and the value should be set between 0 (completely transparent) and 100 (completely opaque).

You can find the sample code to generate a customized Banner App wall below:

```
- (void)viewDidLoad {
    [super viewDidLoad];
    CGRect adframe = CGRectMake(0.0, 0.0, 320.0, 100.0);
    self.adView = [[AvazuADView alloc] initWithFrame:adframe
                                           adType:AVAZU_BANNER_APPWALL
                                           sourceID:@"1234"];

    self.adView.delegate = self;
    [self.view addSubview:self.adView];
    self.adView.isNeedSize = 0;
    self.adView.isNeedReviewNumber = 0;
    self.adView.appCount = 4;
    [self.adView loadAD];
}
```

3.2 Creating AvazuADView by Interface Builder

You can also create [AvazuADView](#) in your [UIViewController](#) by Interface Builder as follows:

1. In xib or storyboard, create an [UIView](#) with proper frame size to fit the [adType](#) you choose.
2. In the identity inspector in the top right corner, set Custom Class to [AvazuADView](#).
3. Open assistant editor, create an IBOutlet instance for the view in your [viewController.h](#) file through Ctrl Drag.
4. Setting constraint for the view through auto layout.
5. Load ad by adding following code in your [viewController.m](#) file:

```
- (void)viewDidLoad {
    [super viewDidLoad];
    self.adView.sourceID=@"6395";
    self.adView.adType = AVAZU_SINGLE_BANNER;
    self.adView.delegate = self;
    [self.adView loadAD];
}
```

Note: For creating [AvazuADView](#) by Interface Builder, it will call init method automatically, so before call method [loadAD](#), you must set [adType](#) and [sourceID](#), or it will fail to load the ad.

You can still customize your ad as we discussed in Chapter 3.1.3.

4. Implementing the delegate

If you need ad status callbacks, implement the delegate property of the AvazuADView. The user can perform necessary action on receiving callback.

Implement the required delegate for ad success and failure methods, as shown below:

```
- (void)avazuADViewLoadAdSucess:(AvazuADView *)adview
{
    NSLog(@"avazuADViewLoadAdSucess");
}

- (void)avazuADView:(AvazuADView *)adview didFailToReceiveAdWithError:(NSError *)error
{
    NSLog(@"avazuADViewLoadAdFail");
    NSLog(@"error:%@", error);
}
```