iOS mOcean SDK

Developer Guide

For iOS SDK Version 3.0

Table of Contents

Table of Contents 2

What’s New in 3.0 3

Implementation Changes 3

System Requirements 3

Feature List 4

Installing the Ad SDK 5

MASTAdView Use Case 7

Simple Ad Integration 8

Interstitial Ad Integration 10

MASTAdView Customization 11

MASTAdView Interstitial Ad Customization 12

Ad Content Customization 12

Content Updates 12

Troubleshooting 13

# What’s New in 2.12

* New code base
* Support for MRAID 2

# Implementation Changes

* Implementations that use contentAlignment or injectionBodyCode must now use the injectionHeaderCode property to apply customization to the viewport or content style.
* Using the device unique identifier from UIDevice has been deprecated by Apple. To send a unique ID create one within the application and set the udid property on MASTAdView.

# System Requirements

* Intel based Mac
* Xcode 4.5 or higher
* iOS 4.3 or higher\*

\*iOS 4.0 and higher can be attempted with previous versions of Xcode and the iOS SDK.

# Feature List

* **Rich media**

SDK supports MRAID 1 and MRAID 2.

* **HTML/ JS ads**

SDK supports displaying web ads using UIWebView component.

* **Image/Text ads**

SDK supports displaying image and text ads with non-UIWebView native components.

* **Location auto detect**

SDK can automatically detect user location.

* **User-Agent auto detect**

SDK automatically detects device User-Agent.

* **Internal browser**

SDK contains built-in browser for displaying ads in application.

* **Ad visibility tracking**

SDK automatically detects ads visibility for controlling updates.

* **Logging**

SDK supports logging.

# Installing the Ad SDK

## Add the SDK library project

Add the MASTAdView.xcodeproj to your projects files. This can be done by dragging it from Finder and dropping into a project group or by using the Xcode’s File>Add Files dialog.

## Add frameworks and libraries

Add the following frameworks and libraries to you project:

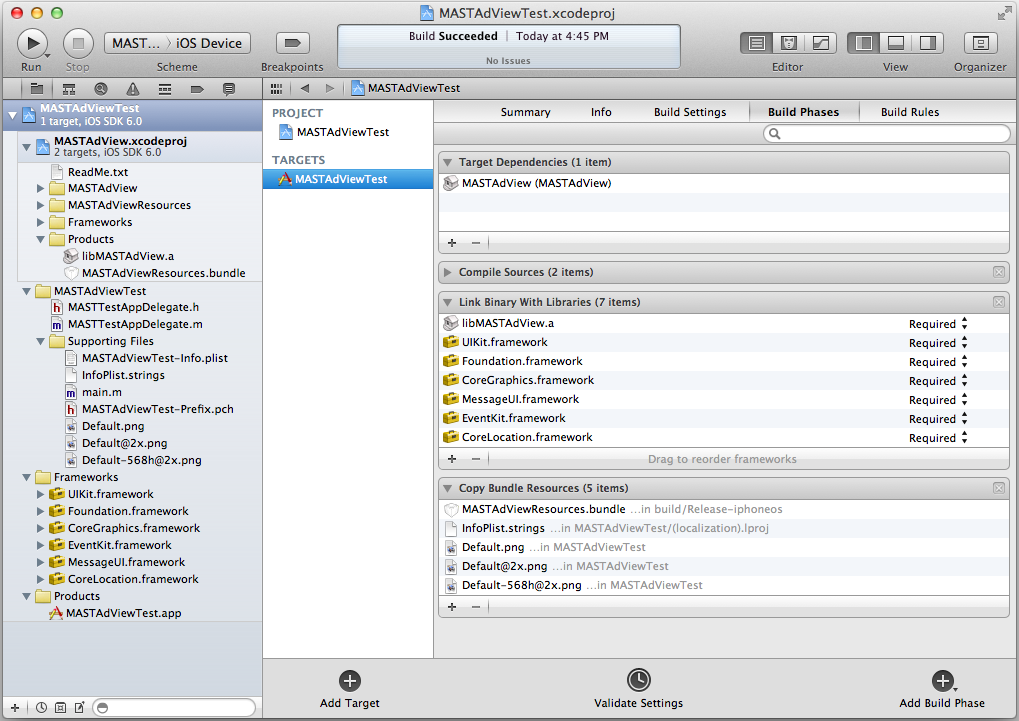
* Foundation
* CoreFoundation
* UIKit
* EventKit
* MessageUI
* CoreLocation
* libMASTAdView.a

## Add resource bundle

Expand the MASTAdView.xccodeproj that was added earlier and it’s Products group. Drag the MASTAdViewResources.bundle to the Copy Files build phase of project target that will include the SDK.

## Set target dependencies

Add the MASTAdView dependency to the Target Dependencies build phase of the project target that will include the SDK.



# MASTAdView Use Case

1. A simple banner that is used for the embedding of the existing form and occupies a small area in it.
2. Interstitial is used for the full screen banner display. It’s displayed only once and after closing it becomes inactive.
   1. Full screen banner is shown as a pop up on top of the main screen of the window and blocks all user’s actions with the application.
   2. Full screen banner is shown as a pop up on top of the main screen of the window and doesn’t block all user actions with the application. As an example, the user can switch tabs.
   3. If several displays are switched on, the banner will be displayed on the main screen only.

# Simple Ad Integration

To add MASTAdView into your application just import MASTAdView.h.  
#import “MASTAdView.h”

Implementation can vary depending on existing code style and layout as well as non-ARC and ARC based projects. Because the MASTAdView can have a delegate it’s good practice to retain a reference and release it in dealloc and not rely on a superview to maintain its lifetime.

In the interface directive add a property to hold the ad view reference:  
@property (nonatomic, strong) MASTAdView\* adView;

In the implementation directive synthesize the new property:  
@synthesize adView;

To ensure proper resource handling, add the following to dealloc to reset the delegate, cancel any timers and release the adView:  
- (void)dealloc  
{  
 [self.adView setDelegate:nil];  
 [self.adView cancel];  
 self.adView = nil;  
  
 // For non-ARC:  
 [super dealloc];  
}

Finally, in the implementation create and add the ad view to a superview:  
- (void)viewDidLoad  
{  
 [super viewDidLoad];  
  
 if (self.adView == nil)  
 {  
 CGRect frame = CGRectMake(0, 0, 320, 50);  
 self.adView = [[MASTAdView alloc] initWithFrame:frame];  
 self.adView.site = @”19829”;  
 self.adView.zone = @”88269”;  
 }  
  
 [self.view addSubview:self.adView];

[self.adView update];  
}

See Also:

* For more code samples examine the Samples application.



If your application supports rotation and the ad view has been configured to be resized (ex: wider when in landscape) simply call update again when notified of the rotation.

- (void)didRotationFromInterfaceOrientation(UIInterfaceOrientation)fromInterfaceOrientation toInterfaceOrientation:(UIInterfaceOrientation)toInterfaceOrientation

{

// If using an update interval call updateWithTimeInterval: again.

[self.adView update];

}

# Interstitial Ad Integration

Interstitial ads work much like inline/banner ads except that they are directly displayed and do not need to be added to a superview. Normal interstitial ads are full screen and modal in appearance. The MASTAdView can be initialized to display interstitial content directly to the screen without having to manage a separate modal view controller.

Interstitial init method.  
self.adView = [[MASTAdView alloc] initInterstitial];

Interstitial control methods.  
[self.adView showInterstitial];  
[self.adView closeInterstitial];

Note that interstitial ad views still require update to be called and can be customized like inline/banner ads.

Since ads can be highly customized it is also to use the ad in inline mode to display partial screen interstitials. In this manner the ad view would be created as normal and the interstitial methods would not be used.

See Also:

* For more code samples examine the Samples application.

# MASTAdView Customization

## Customize view appearance

1. Ad links are opened in Safari by default. To enable the internal browser set the useInteralBrowser property to YES.
2. Default UIView customization such as animation, background color, orientation/sizing masks, etc. can be used on the MASTAdView.
3. The MASTAdView instance allows direct access to the ad content container views. These views can be customized but should not have properties adjusted that would affect their behavior in the MASTAdView view.



# MASTAdView Interstitial Ad Customization

## Customize view appearance

1. Custom close buttons can be used to match the theme of the integrating application.
2. Interstitial ads can be automatically closed after a given time interval.

# Ad Content Customization

## Content filtering

The content can be filtered by setting the various content filtering properties for keywords, city, carrier, dma, etc. See the Mocean Ad Server wiki page for more information.

## Location support

The latitude and longitude properties can be set manually or can be set automatically by the SDK based on the location services available to the device.

If the SDK services are used for location the user will be prompted for location permission. See the Location Detection Configuration section of the API documentation for usage of the SDK location services.

# Content Updates

MASTAdView updates content only by the following methods:

1. Calling the update method. Use this after initializing and during display of the owning view controller.
2. Calling the updateWithTimeInterval: method. This method will cause the SDK to update every interval seconds. Interacting with the current ad suspends the timer. This can be due to a user expanding the ad, clicking and viewing publisher content with the internal web browser or if the user’s action leaves the application.

Call the cancel method to stop ad loading and cancel any timers.

Troubleshooting

## Ad content loading issues

1. Verify the specified content zone has ad content.
2. Implement the ad view’s delegate and debug any ad download failure errors.
3. Enable simple test banners by setting the testMode property to YES.