appnexus
www.appnexus.com
28 west 23rd street, floor 4, new york, ny 10010

P (646) 825-6460
F (646) 825-6465
info@appnexus.com
appnexus

Contents

Overview **3**
Pre-requisites for Mediation **3**
Steps: Integrate OAS Mobile SDK for iOS **4**
Steps: Integrate AppNexus Mobile SDK for iOS **9**
Settings Required in XCode **16**
Enabling Mediation **19**
Setting Targeting parameters for Mediated Ads (Optional) **21**
Setting Location Targeting parameter for Mediated Ads (Optional) **21**
Enabling Mediation Dynamically 23
Mediation Libraries with Adapters **24**
Debugging Mediation **25**

P (646) 825-6460
F (646) 825-6465
info@appnexus.com
appnexus

appnexus 28 west 23rd street, floor 4, new york, ny 10010
www.appnexus.com

Overview

Client-side mediation allows publishers to fill remnant ad spots with ads served from AppNexus, as well as multiple 3rd party ad networks. It is initiated by OAS Mobile SDK, which calls AppNexus Mobile SDK to either fetch an ad from AppNexus inventory, or to call out one or more mediated networks/SDKs in a "waterfall"-like process. If OAS Mobile SDK can't retrieve an ad from or through OAS, it will attempt fetching the ad through AppNexus Mobile SDK and iterate over the list of mediated SDKs in the appropriate order. This is repeated until the ad spot is either filled, or none of the mediated network yields an ad.

Reasons to mediate to another SDK include the following:

- To provide better monetization for remnant ad spots.
- To provide access to information such as a unique user ID or the device's operating system, location, or ID.
- Some networks only accept requests from their own SDKs, forcing the app developer to use the network's SDK to access their demand. Mediation through such network's SDK helps overcoming this limitation.

Pre-requisites for Mediation

1. Publisher application – must compile without any errors.
2. OAS Mobile SDK version 2.0.0 or above for iOS.
3. The latest certified version of the AppNexus SDK – see the release notes to find the correct version. All versions of AppNexus Mobile SDK are available at the following link https://github.com/appnexus/mobile-sdk-ios/releases/ for download. Click the ANSDKBinary zip file to download as shown in the screenshot below.
4.

P (646) 825-6460
F (646) 825-6465
info@appnexus.com
appnexus

RC2.13.2

Edit

davidappnexus released this 9 days ago

Bug Fixes

- MS-2212 Allow RTB banners to fit full screen. New BOOL property shouldResizeAdToFitContainer.
- MS-2820 Support multiple sizes in SDK ad request.
- PR-10 Integrate third party PR: project and schema updates per Xcode 8.

Downloads

ANSDKBinary-RC2.13.2.zip	362 MB
Source code (zip)	
Source code (tar.gz)	

S

upported AppNexus Mobile **SDK version 2.1 or above for iOS.**
Supported Mediation libraries and adapters ~~including Amazon, Mopub, Facebook, iAd, Millennial Media and Google Play and their corresponding adapters.~~

~~Please Note: Millennial Media SDK can be downloaded separately after setting up account with them..~~

Steps: Integrate OAS Mobile SDK for iOS

This section provides step-by-step guide to integrate iOS SDK with the publisher's iOS application.

Please note:

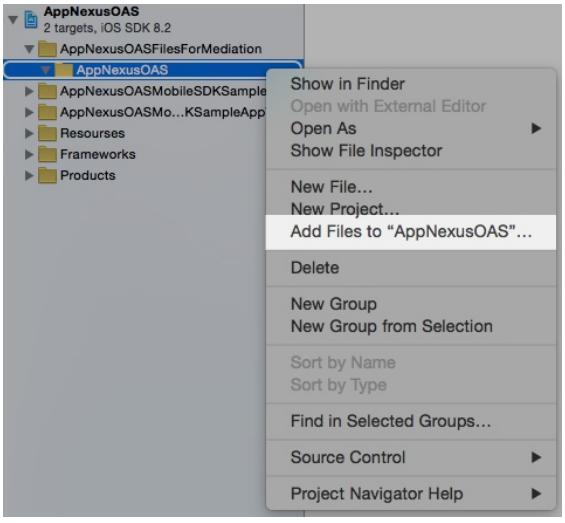
- All the mediation libraries and adapters including AppNexusSDK are optional and should be integrated only when publisher wants to enable mediation.

- Each SDK is linked with the main SDK through its adapter. If publisher is including any particular SDK (for e.g. libMoPubSDK.a) then he/she must include its corresponding adapter (libANSDKMoPubAdapter.a).

Step 1: Unzip OAS Mobile SDK

Step 2: Add include folder to the XCode project

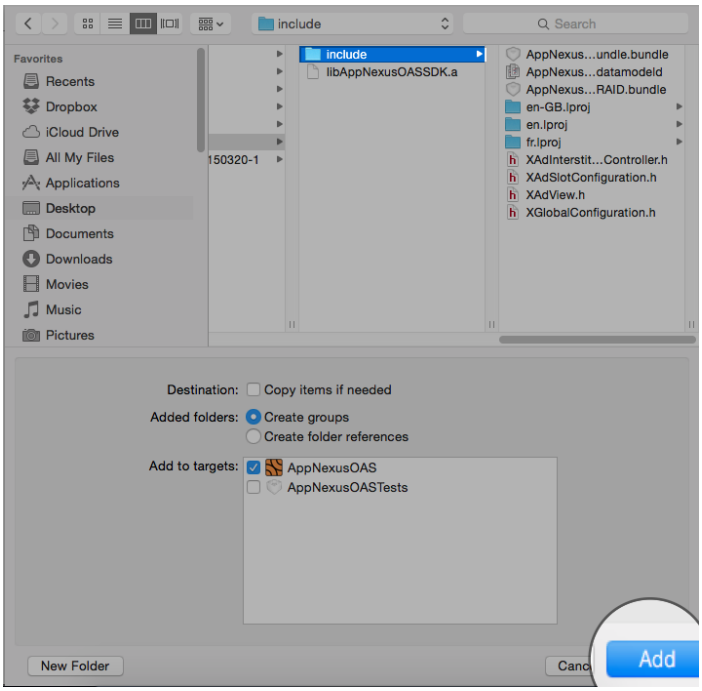
1. Right Click on the folder group inside XCode to which you wish to add the SDK. Click on "Add Files to…"



appnexus 28 west 23rd street, floor 4, new york, ny 10010
www.appnexus.com

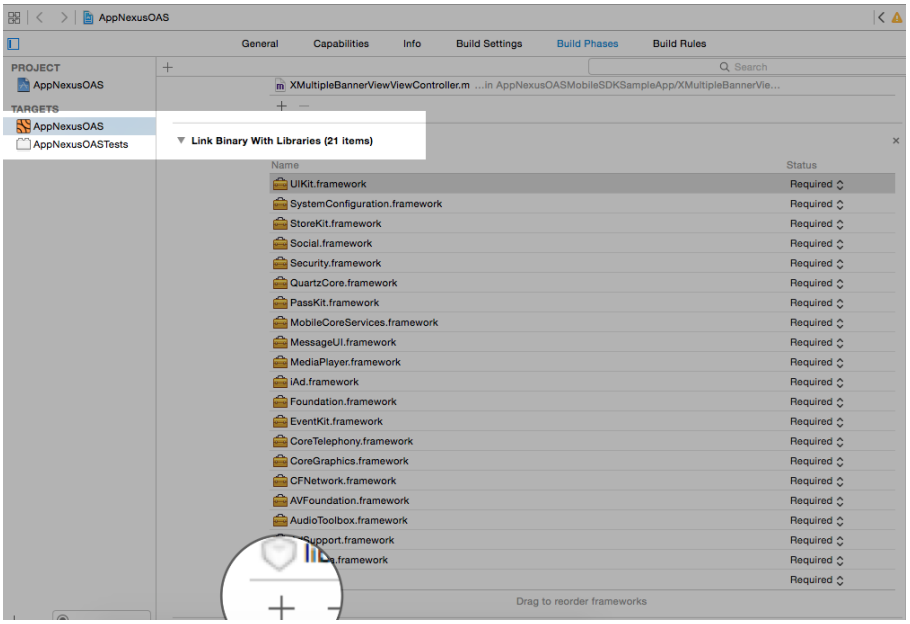
P (646) 825-6460
F (646) 825-6465
info@appnexus.com

2. Browse to the folder containing the unzipped AppNexusOAS SDK files. And select "include" folder to add to the project.

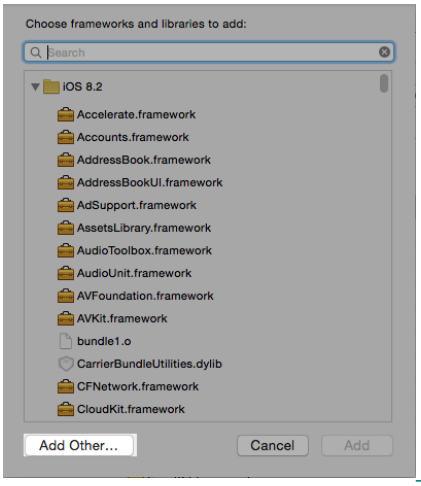


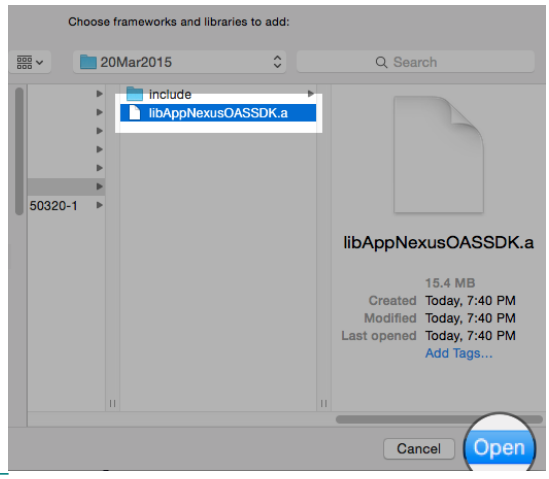
Step 3: Add the OAS Mobile SDK library – libAppNexusOASSDK.a to the XCode project

1. Select the required target, go to build settings and expand “Link Binary with Libraries” section. Click on “+” sign.



2. Click on “Add Other” and Browse to the unzipped AppNexusOAS SDK folder. Select “libAppNexusOASSDK.a” file



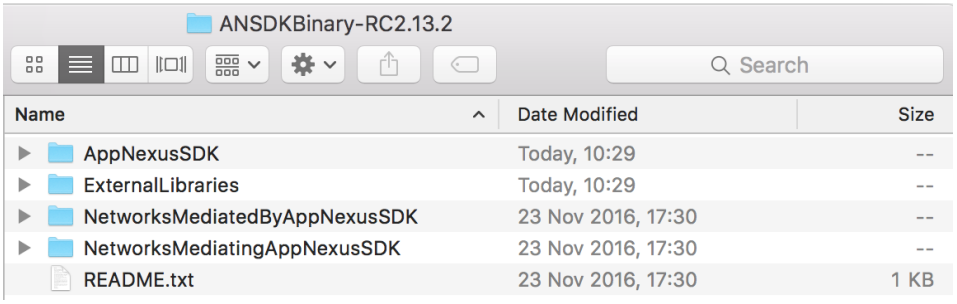


Step 4: Compile the project. Ideally, it should compile without any errors. If any errors are found, please resolve it before going to next steps.

Steps: Integrate AppNexus Mobile SDK for iOS

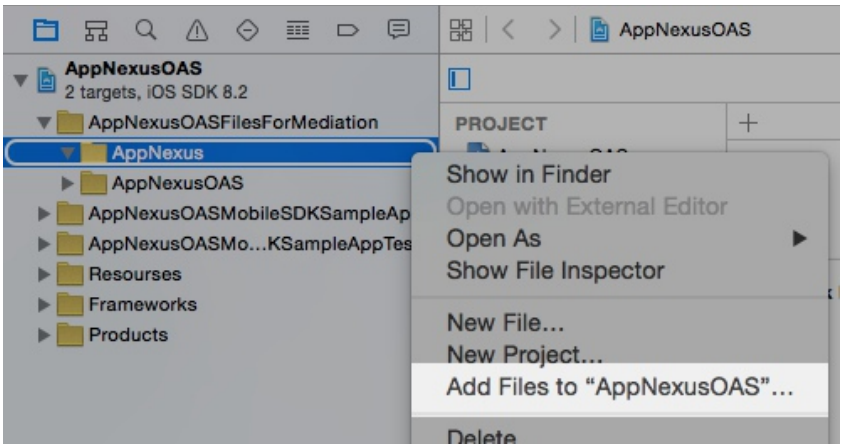
This section provides step-by-step guide to integrate the AppNexus SDK to the publisher's iOS application.

Step 1: Unzip AppNexus SDKDownload the AppNexusSDKBinary zip from the Github releases page. Unzip the binary. You should see 4 folders and the README.txt like the screenshot below.



Step 2: Add the root folder to the XCode Project

1. Right Click on the folder you wish to add the AppNexus SDK to.

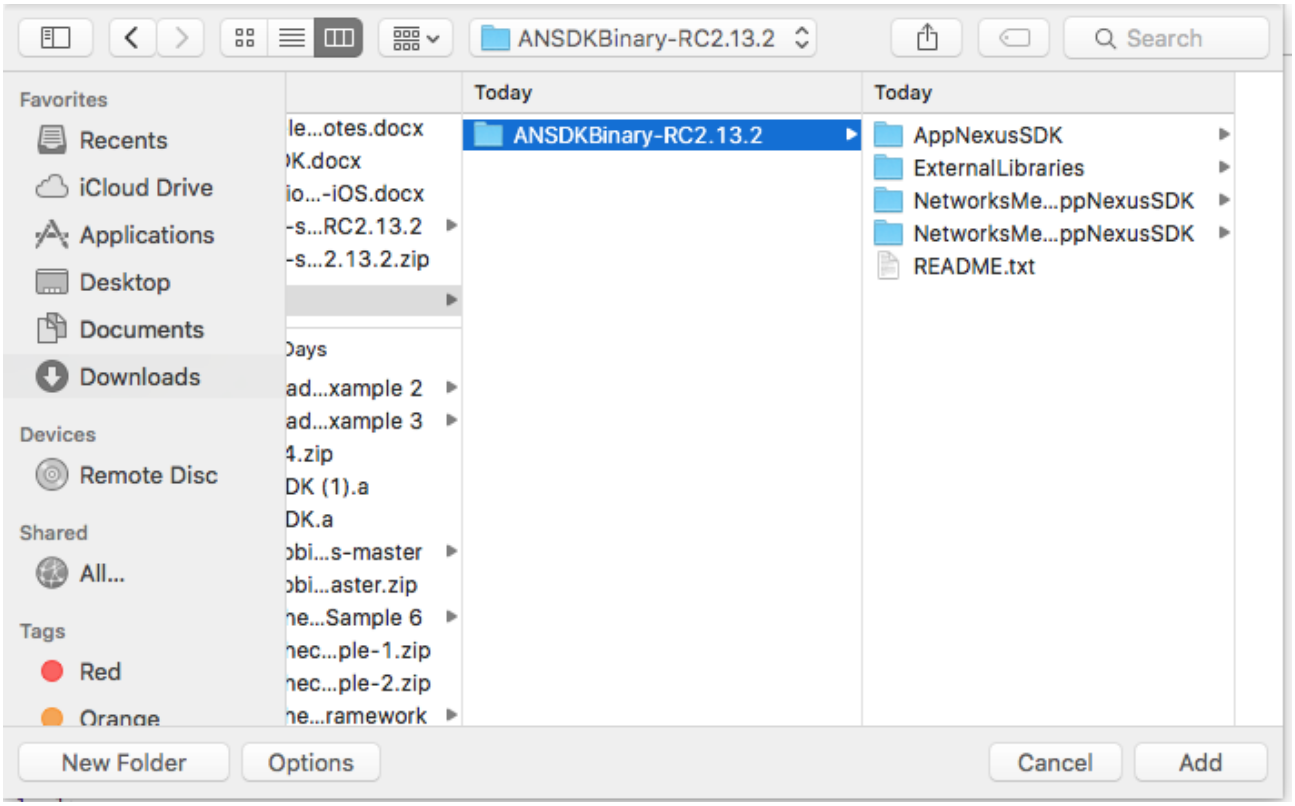


2. Browse to the unzipped folder and select the AppNexus SDK and any additional mediation folders you want to import to your project.

   All the mediation libraries and adapters including AppNexusSDK are optional and should be integrated only when publisher wants to enable mediation. The mediation libraries are located in the External Libraries folder and the mediation adapters are located in NetworksMediatedByAppNexusSDK folder as shown in the screenshot below.

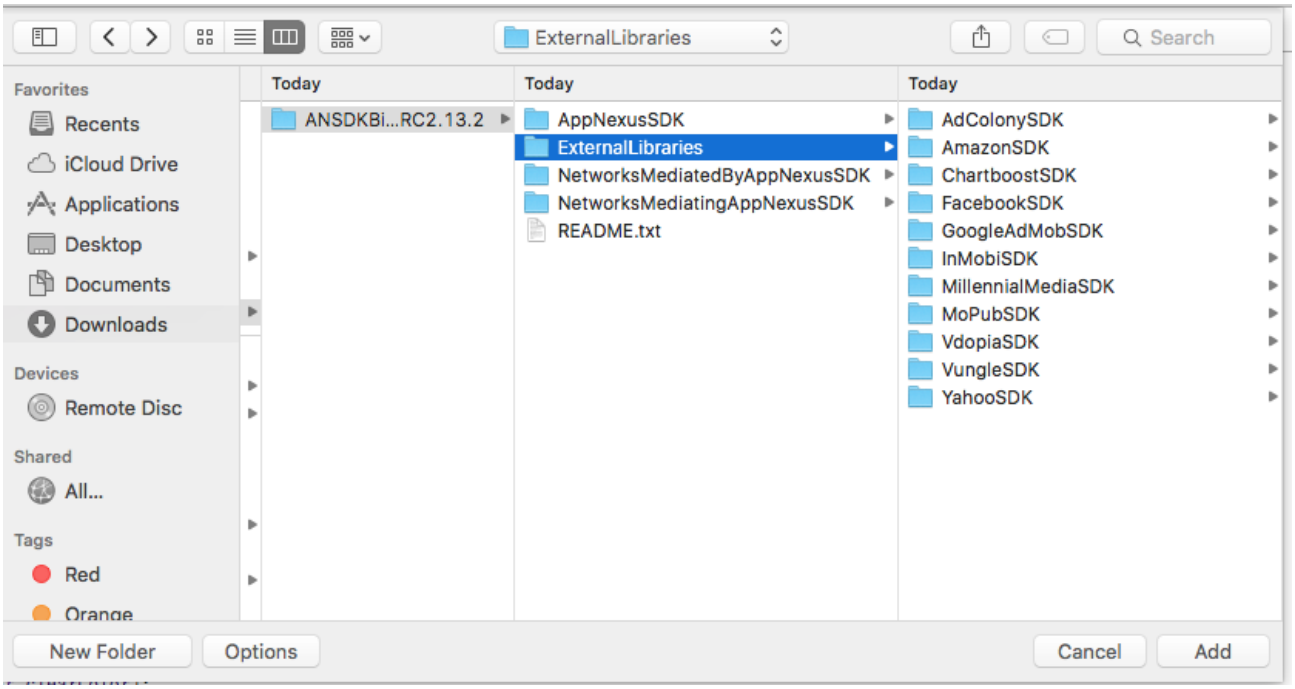
   Each SDK is linked with the main SDK through its adapter. If publisher is including any particular SDK (for e.g. libMoPubSDK.a) then he/she must include its corresponding adapter (libANSDKMoPubAdapter.a).

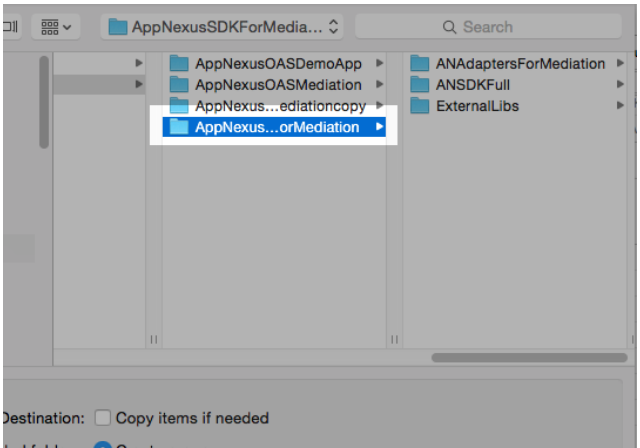
3. Browse to the unzipped folder and select the root folder



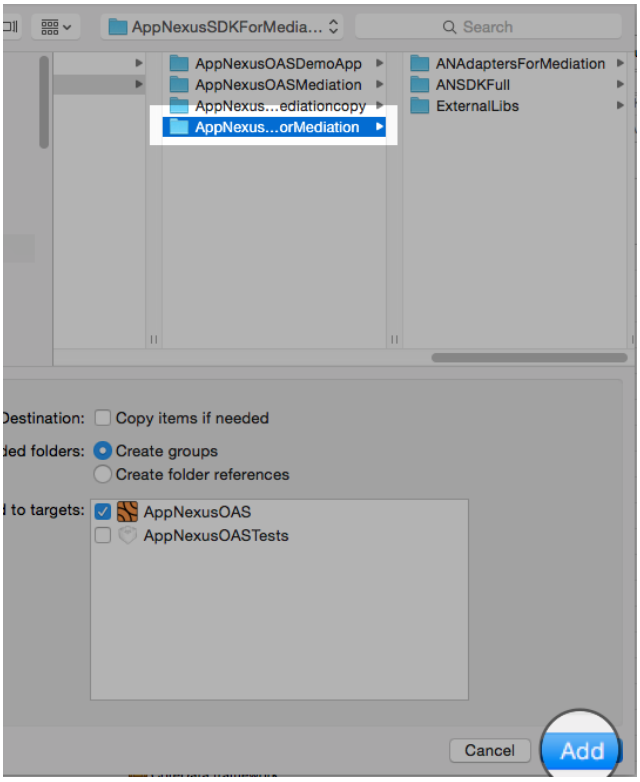
P (646) 825-6460
F (646) 825-6465
info@appnexus.com
appnexus

appnexus 28 west 23rd street, floor 4, new york, ny 10010
www.appnexus.com

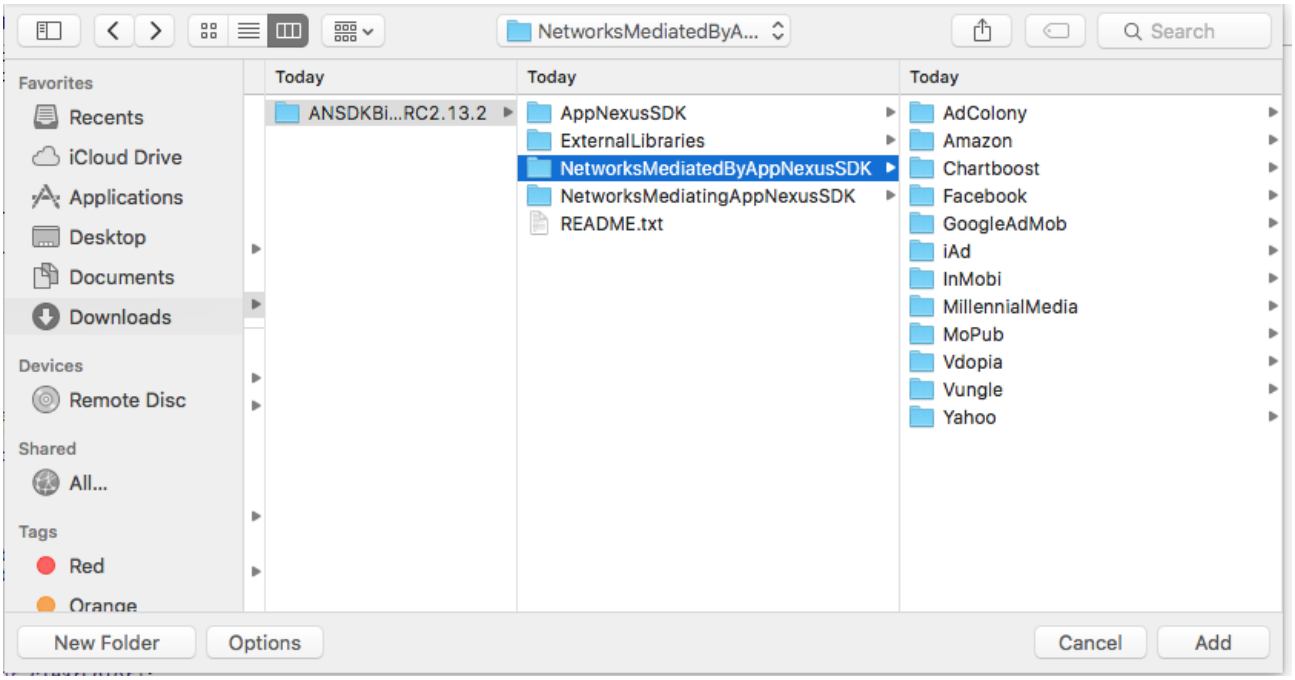




P (646) 825-6460
F (646) 825-6465
info@appnexus.com
appnexus



P (646) 825-6460
F (646) 825-6465
info@appnexus.com
appnexus

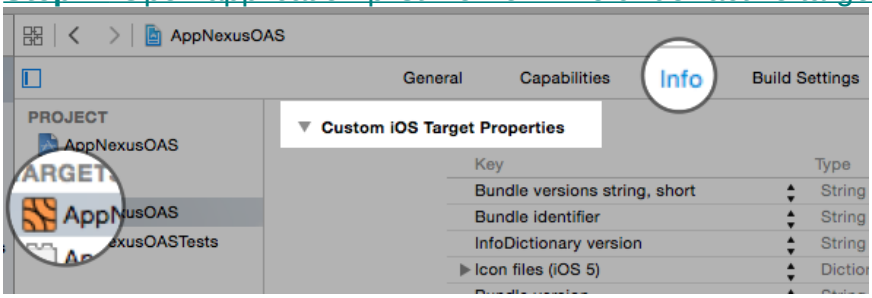


Step 3: Compile the project. Ideally, it should compile without any errors. Please resolve any errors before going to the next steps

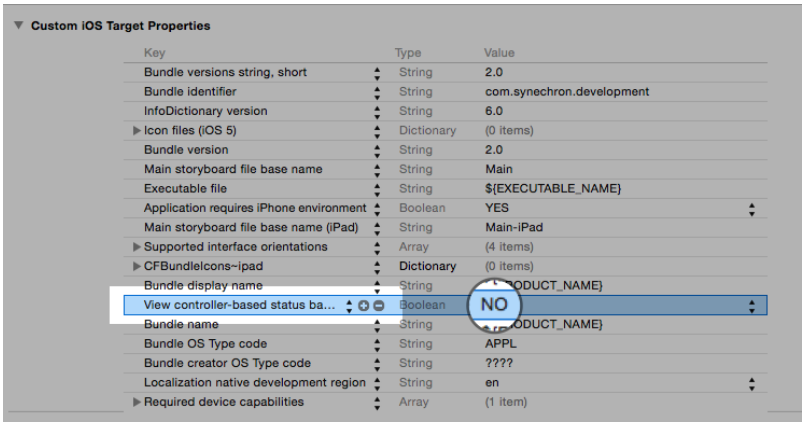
Settings Required in XCode

These settings are required to ensure optimal functioning of the OAS Mobile SDK:

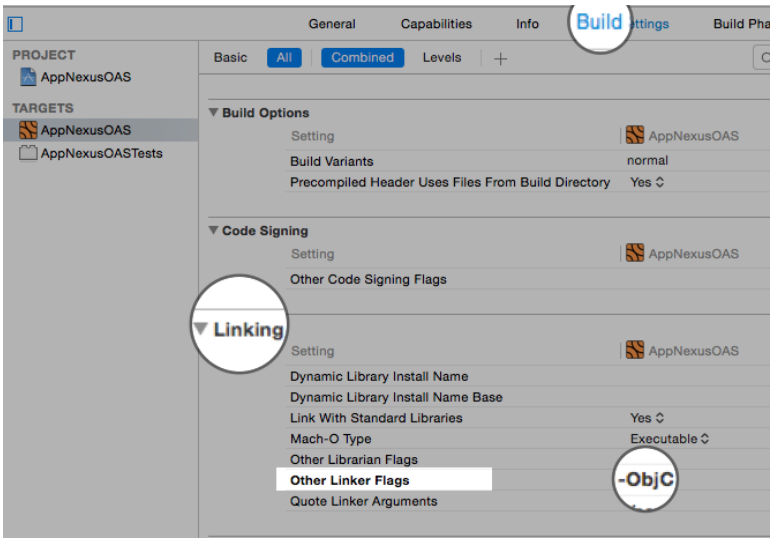
Step 1: Open application plist file from info under active targets.



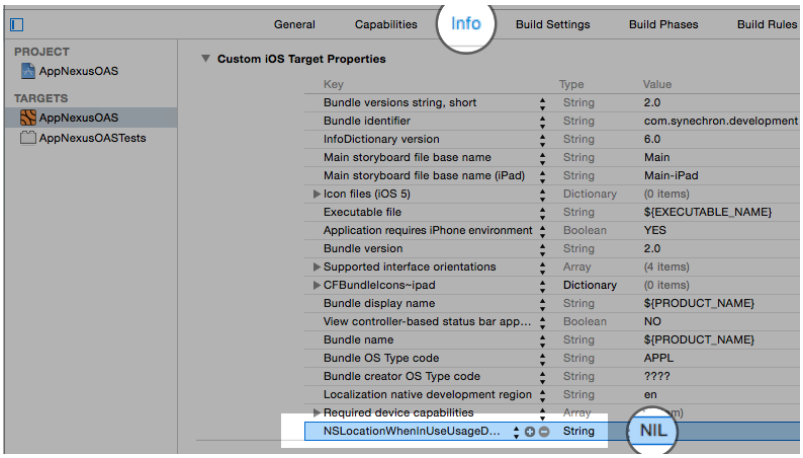
1. Add “View controller-based status bar appearance” to the application plist and set the value to NO.



2. Add “-ObjC” to “Other Linker Flags” under application target build settings



3. Add “NSLocationWhenInUseUsageDescription” to the application plist and leave the value as NIL.



appnexus 28 west 23rd street, floor 4, new york, ny 10010
www.appnexus.com

P (646) 825-6460
F (646) 825-6465
info@appnexus.com

appnexus 28 west 23rd street, floor 4, new york, ny 10010
www.appnexus.com

Enabling Mediation

With OAS Mobile SDK, mediation can be enabled at the application level or at individual slot levels.

When mediation is enabled at application level, it will be enabled throughout the application. Meaning, if any of the banners or interstitial (except Pre-roll) fails to receive ad from OAS, it would try to fill the ad slot from AppNexus, or through mediated ad networks. But it requires publishers to set the mediation placement id along width and height parameters (for mediated banner ads) in slot configuration.

On the other hand, when mediation is enabled at slot levels, the mediation is enabled only for that slot. Meaning, if mediation is enabled for a banner ad on screen 3, then OAS Mobile SDK would try and fetch ads from mediation only for that particular banner placement on screen 3 and not for any other placements through out the application. This way, OAS Mobile SDK provides flexibility to the publishers to enable or disable mediation at individual slot levels as well as at application level.

Steps to enable mediation at the application level:

Step 1: Create the object of XGlobalConfiguration and set canMediate property to YES. Or, add the below code snippet when the application start – Ideal place would be the appDelegate's didFinishLaunchingWithOptions method.

```
[XGlobalConfiguration sharedInstance].canMediate = YES;
```

Step 2: At the slot level, set the following parameters mediationPlacementId (NSString), mediationBannerHeight (float), mediationBannerWidth (float). Please refer to the code snippets given below on how to set these parameters.

```
configuration.mediationPlacementId = @"12345";
configuration.mediationBannerWidth = 320.0;
configuration.mediationBannerHeight = 50.0;
```

Please Note:

P (646) 825-6460
F (646) 825-6465
info@appnexus.com
appnexus

A slot level setting for mediation takes precedence over application level setting. This means that even if mediation is disabled at the application level, but enabled for a particular slot, then upon ad request failure from OAS server, SDK would still request the mediated network to fill that slot.

In case if mediation is enabled at the application level, but disabled for a particular slot, then upon a failed request from OAS server, SDK would terminate the flow and not make any further request to mediated ad networks for this slot.

Steps to enable mediation at the slot level:

Step 1: Create the object of XAdSlotConfiguration and set canMediate property to YES.

```
XAdSlotConfiguration *configuration = [XAdSlotConfiguration new];
    configuration.canMediate = YES;
```

Set the following parameters mediationPlacementId (NSString), mediationBannerHeight (float), mediationBannerWidth (float). Please refer to the code snippets given below on how to set these parameters.

Step 2: Setting PlacementId

This is a required parameter and mediation would fail to start without this parameter value set.

```
configuration.mediationPlacementId = @"12345";
```

Step 3: Setting mediationBannerWidth and mediationBannerHeight

This is a required parameter while requesting banner ads. This parameter is not required for requesting interstitial ads.

```
 configuration.mediationBannerWidth = 320.0;
 configuration.mediationBannerHeight = 50.0;
```

Setting Targeting parameters for Mediated Ads (Optional)

Setting Optional targeting parameters (Age, Gender, Custom Keywords & Location)

These parameters can be used for targeting the ad request based on age, gender, custom keywords and/or location.

Setting age

configuration.mediationTargetedAge = 32;

Setting Gender

configuration.mediationTargetedGender = XMediationTargetedGenderFemale;

XMediationTargetedGender is available inside XAdView as an enum.

Setting Custom Keywords

configuration.mediationTargetedKeywords = [NSDictionary dictionaryWithObjects:[NSArray arrayWithObjects:@"mascara", @"eyeliner", nil] forKeys:[NSArray arrayWithObjects:@"Cosmetics", @"Cosmetics", nil]];

Setting Location Targeting parameter for Mediated Ads (Optional)

User location must be set at global level.

[XGlobalConfiguration sharedInstance].mediationTargetedLocation = _userLocation;

where _userLocation is obtained by implementing the below code.

_

Step 1: Import the following framework to the application class

```
#import <CoreLocation/CoreLocation.h>
```

Step 2: Implement the CLLocationManagerDelegate

```
@interface AppNexusOASRichMediaViewController : UIViewController<XAdViewDelegate, CLLocationManagerDelegate>
```

Step 3: declare the required properties

```
@property (nonatomic, strong) CLLocation *userLocation;
@property (nonatomic, strong) CLLocationManager *locationManager;
```

Step 4: Synthesize them

```
@synthesize userLocation = _userLocation;
@synthesize locationManager;
```

Step 5: call the method to obtain location somewhere at the beginning. In this case, we have used it inside viewDidLoad

```
[self getUserLocation];
```

where getuserLocation method is as follows –

```
- (void) getUserLocation{

    locationManager = [[CLLocationManager alloc] init];
    CLAuthorizationStatus authStatus = [CLLocationManager authorizationStatus];

    if (authStatus == kABAuthorizationStatusRestricted || authStatus == kABAuthorizationStatusDenied || authStatus == kABAuthorizationStatusNotDetermined) {
        if ([locationManager respondsToSelector:@selector(requestWhenInUseAuthorization)]) {
            [locationManager requestWhenInUseAuthorization];
        }else if([locationManager respondsToSelector:@selector(requestAlwaysAuthorization)]){
```

```
        [locationManager requestAlwaysAuthorization];
    }
}

locationManager.delegate = self;
locationManager.desiredAccuracy = kCLLocationAccuracyThreeKilometers;
locationManager.distanceFilter = kCLDistanceFilterNone;
[locationManager startUpdatingLocation];

}
```

Step 6: Implement the LocationManagerdelegate didUpdateLocation

```
- (void)locationManager:(CLLocationManager *)manager didUpdateLocations:(NSArray *)locations{

_userLocation = [locations lastObject];
[locationManager stopUpdatingLocation];

}
```

Enabling Mediation Dynamically

To enable mediation dynamically through OAS, publishers can use following XML format as a house ad. Upon receiving this response, SDK (version 2.1.0 and above) will treat it as a no ad response and enable mediation behind the scenes with the updated placement id and other parameters as mentioned in the xml. If app developer has already specified a placement id in SDK's configuration, then placement id mentioned in XML would take precedence.

OAS should serve this XML only when it fails to deliver any ad. Such an ad will be a house ad.

```
<?xml version="1.0" encoding="UTF-8"?>
<MobileSDK>
   <ClientSideMediation placementId="12345" width="320" height="50" />
</MobileSDK>
```

Note:

- AdType attribute for the above XML must be set as text/xml for SDK to detect correctly.
- "width" and "height" attributes are optional, as it is required only by the banner ads. For interstitial ads width and height is diligently ignored. Also, since XML is case sensitive, it is important to maintain the letter cases as it is. Only the values for placementId, width and height are allowed to vary.

Mediation Libraries with Adapters

AppNexus OAS SDK supports 5 different types of mediated ad networks. Following is the list of supported ad networks with their library and corresponding AppNexus adapter.

1. **Amazon**
   - Library name: AmazonAd.framework
   - Adapter name: libANSDKAmazonAdapter.a
2. **Facebook**
   - Library name: FBAudienceNetwork.framework
   - Adapter name: libANSDKFacebookAdapter.a

3. **Mopub**
    - Library name: libMoPubSDK.a
    - Adapter name: libANSDKMoPubAdapter.a

4. **AdMob/Google Play/DFP**
    - Library name: GoogleMobileAds.framework
    - Adapter name: libANSDKGoogleAdMobAdapter.a

5. **Millennial Media**
    - Library name: MillennialMedia.framework
    - Adapter name: libANSDKMillennialMediaAdapter.a

Please Note: Millennial Media SDK can be downloaded separately after setting up account with them.

Debugging Mediation

LogMessages:

Failed to retrieve ad from OAS. Cannot mediate, as required delegate xAdInterstitialDidLoad is not implemented.
While trying to fill the slot with mediated ads, SDK realized that the required delegate implementation for showing interstitial ad is missing. To show the mediated interstitial ad, this delegate is a must to avoid any unintended behavior within the SDK.

LogMessage:

Failed to retrieve ad from OAS. Cannot mediate for Pre-roll ads.
Tried to fill the pre-roll slot with ad via mediation, however, mediation is currently not supported for pre-roll ad.

appnexus 28 west 23rd street, floor 4, new york, ny 10010
www.appnexus.com

LogMessage:

Cannot mediate. Please enable mediation at slot level or application level.
To fill the slot with ad via mediation, you need to enable mediation at the application level or at the slot level.

LogMessage:

Cannot Mediate. PlacementId not set.
To fill slots via mediation, placementId needs to be set at the slot configuration.

LogMessage:

Cannot Mediate. BannerWidth or BannerHeight not set.
To fill banner slots via mediation, mediationBannerWidth and mediationBannerHeight needs to be set at the slot configuration.

LogMessage:

Missing required mediation classes.
Publisher does not integrate the mediation library. To fill the slot with mediated ads, the AppNexus library along with the supported header files are required.

P (646) 825-6460
F (646) 825-6465
info@appnexus.com
appnexus