Freestar Ads Mediation Unity iOS

Last Modified on 05/31/2022 11:22 am EDT

Freestar Ads Mediation - Unity Integration Guide for iOS

Supported Ad Partners

Note: The supported list for Unity may differ from our Native iOS

Ad Provider	SDK Version	Ad Unit Types
AdColony	4.7.2	Fullscreen Interstitial & Rewarded
AppLovin	11.0.0	Fullscreen Interstitial & Rewarded, Banner 320x50
AppLovinMax	11.0.0	Fullscreen Interstitial & Rewarded, Banner 300x250, Banner 320x50
Criteo	4.3.1	Fullscreen Interstitial & Rewarded, Banner 300x250, Banner 320x50
Admob	9.2.0	Fullscreen Interstitial & Rewarded, Banner 300x250, Banner 320x50, Native
Google Ads Manager	9.2.0	Fullscreen Interstitial & Rewarded, Banner 300x250, Banner 320x50, Native
MoPub (deprecated)	5.18.2	Fullscreen Interstitial & Rewarded, Banner 300x250, Banner 320x50, Native
Tapjoy	12.8.0	Fullscreen Rewarded
Unity Ads	4.1.0	Fullscreen Interstitial & Rewarded
Vungle	6.10.6	Fullscreen Interstitial & Rewarded
Google IMA	3.13.0	Preroll
Nimbus	1.10.5	Fullscreen Interstitial & Rewarded, Banner 300x250, Banner 320x50
TAM (Amazon Publisher Services)	4.3.0	Fullscreen Interstitial & Rewarded, Banner 300x250, Banner 320x50

Pangle	3.7.0.8	Fullscreen Interstitial & Rewarded, Banner 300x250, Banner 320x50, Native
Hyprmx	6.0.0	Fullscreen Interstitial & Rewarded, Banner 300x250, Banner 320x50
Yahoo	1.14.2	Fullscreen Interstitial, Banner 300x250, Banner 320x50

Getting Started

Start displaying Freestar Ads in your Unity game today by following the simple steps below.

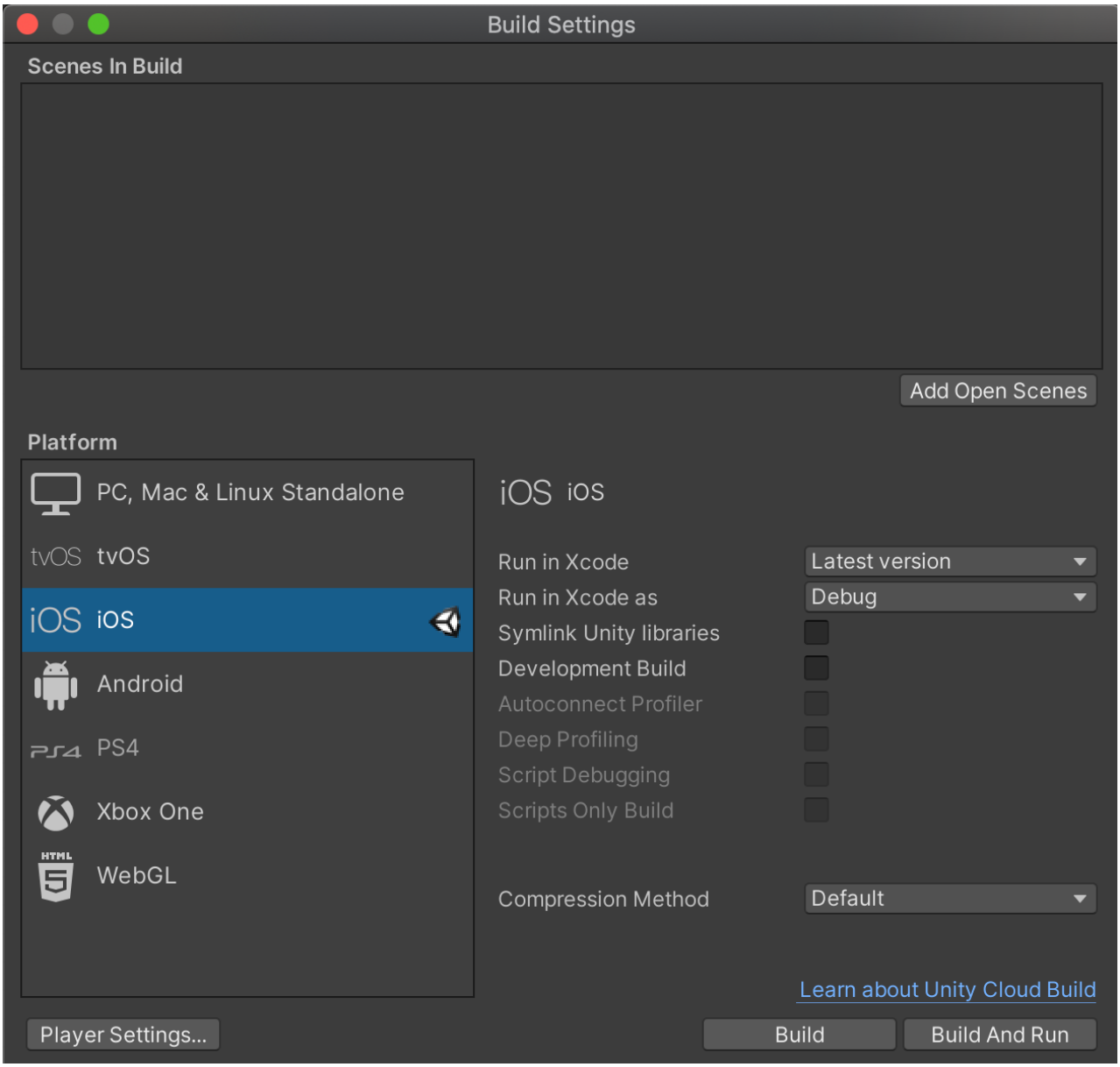
Requirements

- Unity 2020 or newer
- Xcode 13 and iOS 11 or higher
- Cocoapods

Begin

Please follow the instructions in order.

Make sure you switched to iOS:

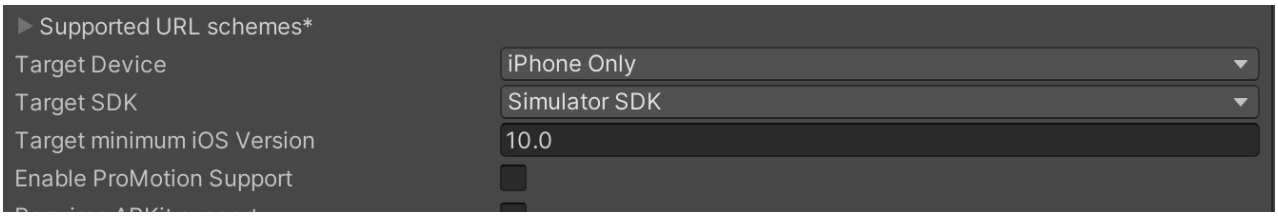


Unity Player Settings

Other settings: Target minimum iOS version: 11.0

Other settings: Graphics API: Open GLES2

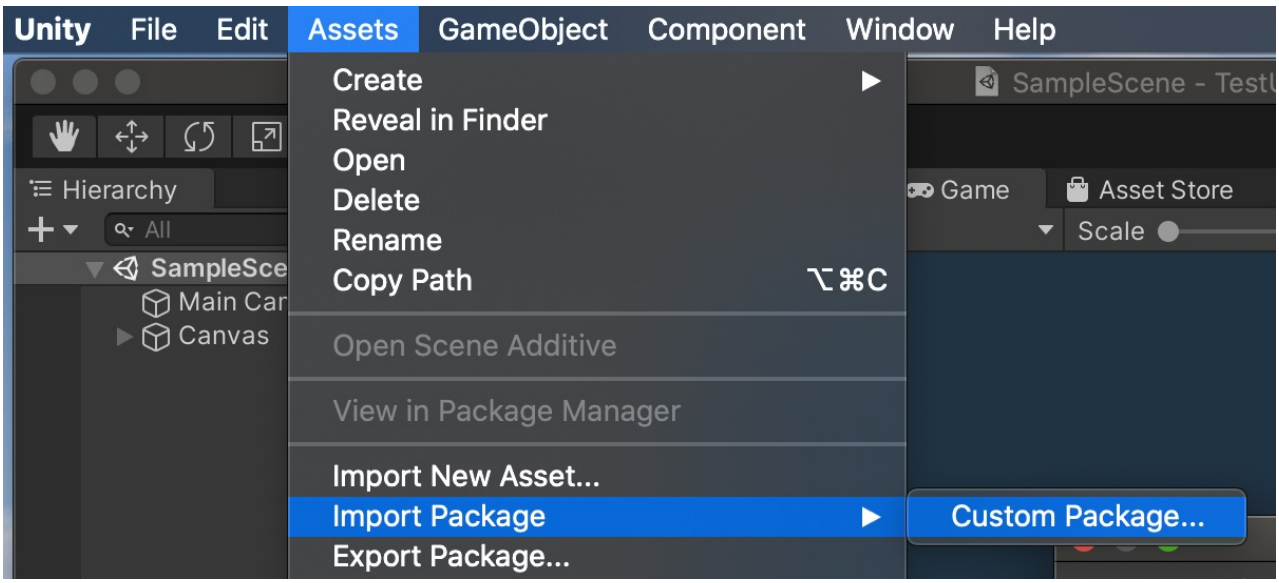
Other settings: Target SDK: Either **Simulator SDK** or **Device SDK** Unity iOS projects are generated for one or the other, but not both.



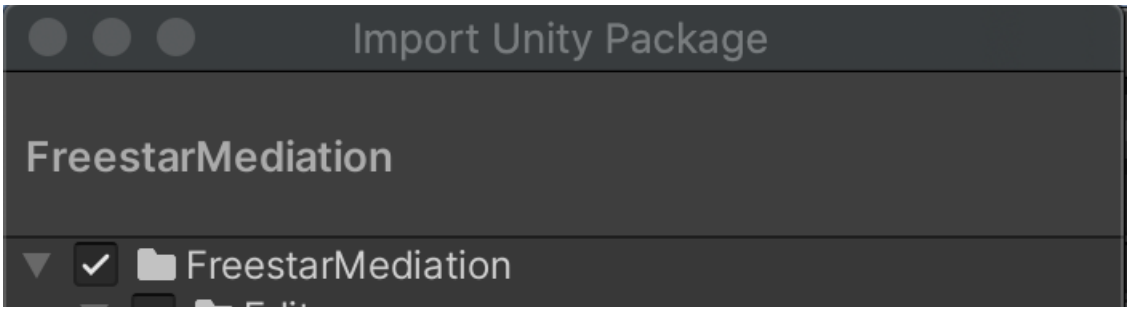
Import Custom Package

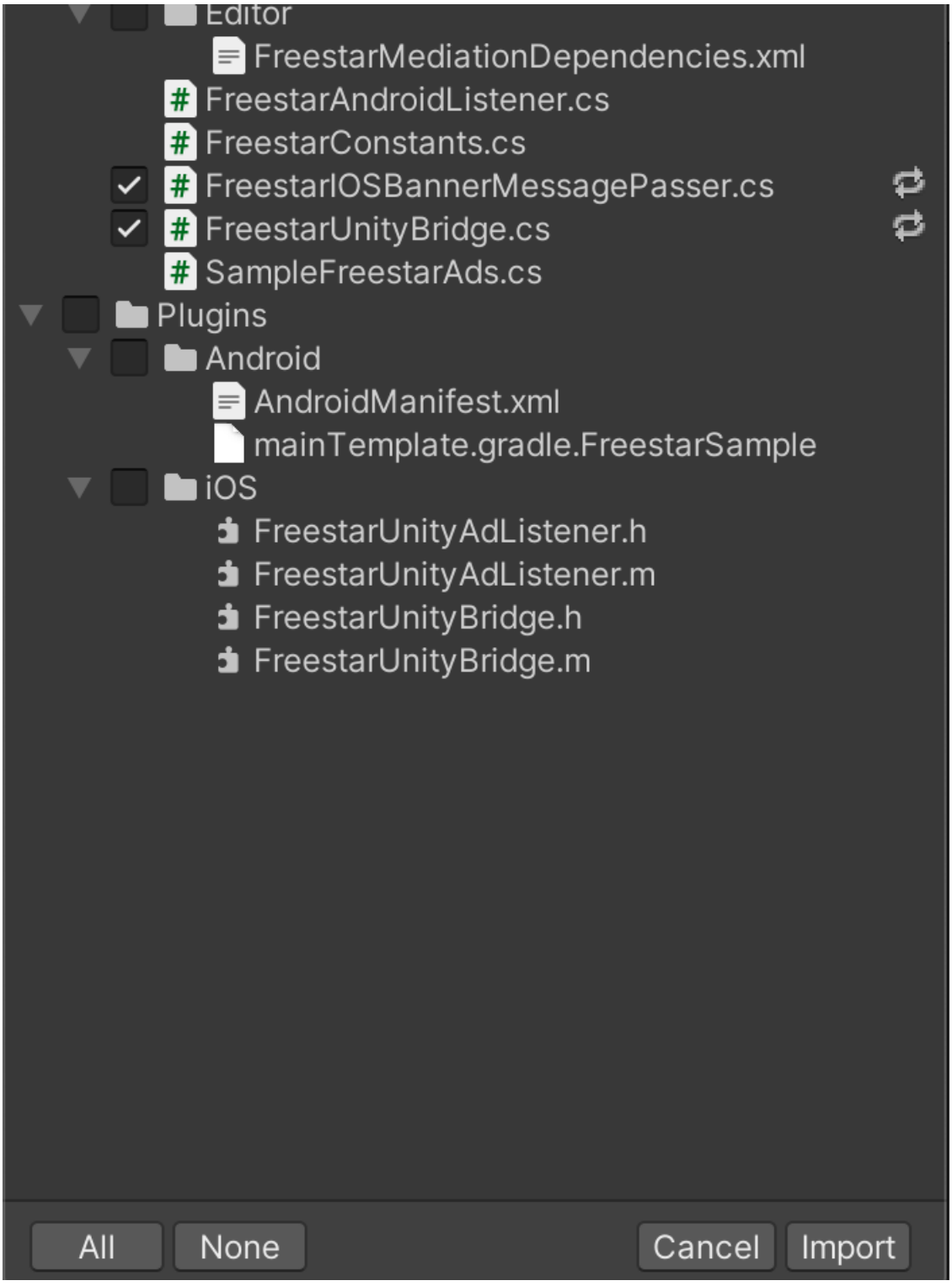
Download FreestarMediation.unitypackage

Next, we will import it into Unity as a Custom Package.



Assets: Import Package: Custom Package





Hit the Import button.

Additional Post Setup Steps

For certain project file type(s), it may be necessary to make manual changes to your Xcode project

to make it build correctly.

1. Add this line in the pod file and save it. Before saving you can exclude (by removing the pod dependencies) any mediation partners that you don't want to integrate (Please contact your account manager for more details):

   ```
   Podfile:
   use_frameworks! :linkage => :static
   ```

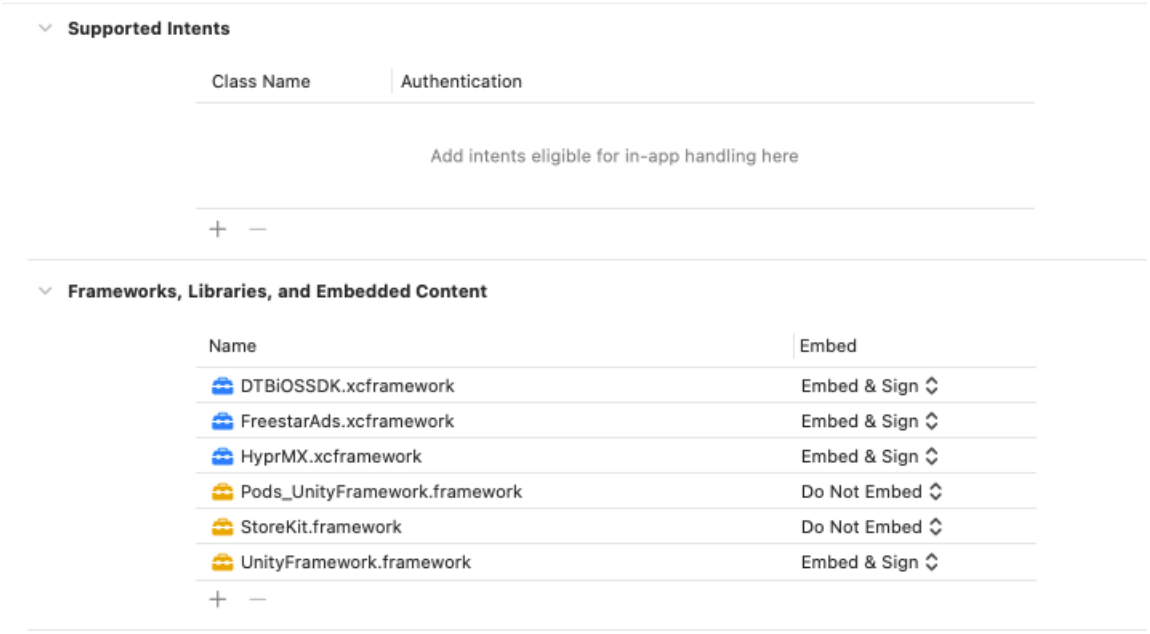
2. Run `*pod install —-repo-update*` to download and install the latest version.
3. Open the `*.xcworkspace*` file by double clicking it.
4. Go to:

   Pods → Target Support Files → Pods-UnityFramework → Pod-UnityFramework-resources.sh

   And delete below lines:

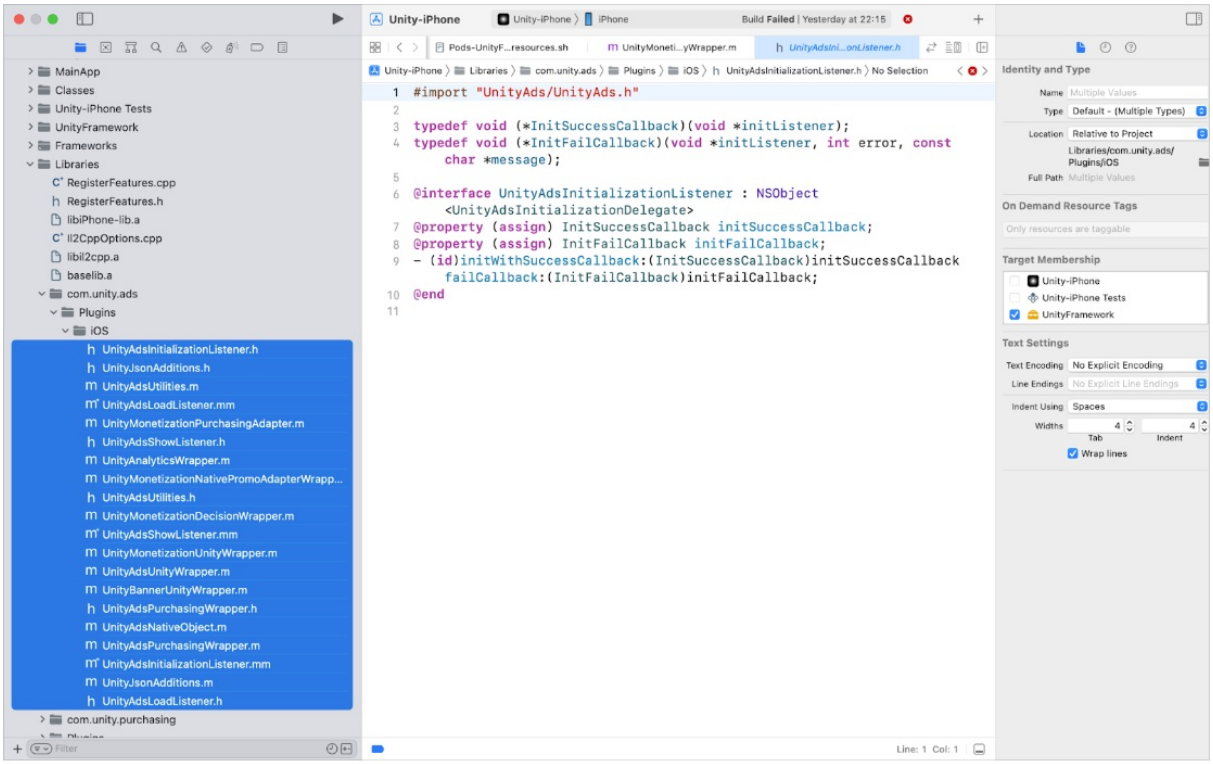
   ```
   install_resource "${BUILT_PRODUCTS_DIR}/FreestarAds/FreestarAds.framework/Freestar-Native-Medium.nib"
   install_resource "${BUILT_PRODUCTS_DIR}/FreestarAds/FreestarAds.framework/Freestar-Native-Small.nib"
   ```

   There could be more than one instance of these lines depending on the number of "$CONFIGURATION". Please make sure to delete these two lines from all configurations.
5. Add `*/usr/lib/swift*` to your Run Search Paths Xcode build setting.
6. Add and embed below shown frameworks.



7. Go to -> Libraries -> com.unity.ads -> plugin -> iOS

   Select all files under this and deselect the 'UnityFramework' under target membership on right hand side.



Resolve iOS Dependencies

One of the files imported in the `FreestarMediation` package is the dependencies spec: `Editor/FreestarMediationDependencies.xml`. This will be used to generate the Podfile for the Xcode project. The relevant section is the following:

```
<iosPods>
  <iosPod name="FreestarAds" version="~> 5.0"></iosPod>
  <iosPod name="FreestarAds-AdColony" version="~> 4.7"></iosPod>
  <iosPod name="FreestarAds-AppLovin" version="~> 11.0"></iosPod>
  <iosPod name="FreestarAds-AppLovinMax" version="~> 11.0"></iosPod>
  <iosPod name="FreestarAds-Tapjoy" version="~> 12.8"></iosPod>
  <iosPod name="FreestarAds-Googleadmob" version="~> 9.2"></iosPod>
  <iosPod name="FreestarAds-Googleadmob/Facebook" version="~> 9.2"></iosPod>
  <iosPod name="FreestarAds-GAM" version="~> 9.2"></iosPod>
  <iosPod name="FreestarAds-GAM/Facebook" version="~> 9.2"></iosPod>
  <iosPod name="FreestarAds-TAM" version="~> 4.3"></iosPod>
  <iosPod name="FreestarAds-Criteo" version="~> 4.3"></iosPod>
  <iosPod name="FreestarAds-Unity" version="~> 4.1"></iosPod>
  <iosPod name="FreestarAds-Vungle" version="~> 6.10"></iosPod>
  <iosPod name="FreestarAds-Pangle" version="~> 3.7"></iosPod>
 </iosPods>
```

You may wish to only include some ad partners. Remove or comment out the lines listing those you do not wish to use. However, the `FreestarAds` pod is required.

Freestar Initialization and Displaying Ads

Initialization

Within your Unity app, you need to initialize the Freestar SDK. After you imported the `FreestarMediation` package, you can use the C# code within by importing the Freestar namespace:

```
import Freestar;
```

Now you will be able to call methods in Freestar classes, the most important of which is the `FreestarUnityBridge`. You will need a Freestar API Key, and pass it to the initializer. For efficiency, it is best to do this as soon as possible within your app's lifecycle.

```
FreestarUnityBridge.initWithAPIKey(YOUR_API_KEY);
```

This will initialize the SDK and enable it to run ads. The Freestar Unity plugin supports two types of fullscreen ads: interstitial and rewarded, and banner ads in two sizes: 320x50 display points, and 300x250 display points.

Interstitial Ads

Interstitial ads take over the app, running in fullscreen, and allow return to the app upon completion. You can opt in to receive callbacks when key events occur in an ad's lifecycle, allowing you to make any changes necessary (such as pausing your game while the ad is playing). To do this, your Unity code needs to implement the `FreestarInterstitialAdCallbackReceiver` interface:

```
void onInterstitialAdLoaded(string placement);
void onInterstitialAdFailed(string placement);
void onInterstitialAdShown(string placement);
void onInterstitialAdClicked(string placement);
void onInterstitialAdDismissed(string placement);
```

And then pass in the object that implements this interface to `FreestarUnityBridge`:

```
FreestarUnityBridge.setInterstitialAdListener(this);
```

Finally, you request to load an ad:

```
FreestarUnityBridge.loadInterstitialAd (AD_PLACEMENT);
```

Note: there are additional detailes of what placements are and the format they need to be in our guide for native app integration. If you do not wish to use placements, simply pass in an empty string:

```
FreestarUnityBridge.loadInterstitialAd ("");
```

While implementing callbacks is optional, showing an ad before it's loaded will not work. The `onInterstitialAdLoaded` callback informs your unity code when the ad is ready. You can now make the following call:

`FreestarUnityBridge.showInterstitialAd (AD_PLACEMENT);`

Make sure the placement strings match, if you are using more than one placement. The ad will now take over the app, and make the appropriate callbacks; the `onInterstitialAdDismissed` is sent when the ad returns control back to the app.

Rewarded Ads

Rewarded ads behave like interstitial ads, but are augmented with an incentivization mechanism. Typically, the user is rewarded by in-app virtual currency upon completion of the ad. The currency and amount are under your control, and thus need to be passed to `FreestarUnityBridge`.

To do this, your Unity code needs to implement the `FreestarRewardedAdCallbackReceiver` interface:

```
void onRewardedAdLoaded(string placement);
void onRewardedAdFailed(string placement);
void onRewardedAdShown(string placement);
void onRewardedAdFinished(string placement);
void onRewardedAdDismissed(string placement);
```

And then pass in the object that implements this interface to `FreestarUnityBridge`:

`FreestarUnityBridge.setRewardedAdListener(this);`

Finally, you request to load an ad:

`FreestarUnityBridge.loadRewardedAd (AD_PLACEMENT);`

While implementing callbacks is optional, showing an ad before it's loaded will not work. The `onRewardedAdLoaded` callback informs your unity code when the ad is ready. You can now make the following call:

```
FreestarUnityBridge.showRewardedAd (
  AD_PLACEMENT, //use "" if you don't need multiple placements
  50,          //reward amount
  "Coins",     //reward name
  "Freestar1"   //User ID, use "" if not utilized
  "SECRET_KEY"  //An optional secret for greater security, use "" if not utilized
);
```

Make sure the placement strings match, if you are using more than one placement. The ad will now take over the app, and make the appropriate callbacks; the `onRewardedAdDismissed` is sent when the ad returns control back to the app.

The `onRewardedAdFinished` is called *before* the dismissed callback, and may be called when the ad is

still playing. It is an indicator that the reward for the ad may now be given to the app user.

Banner Ads

Banner ads are displayed on top of the app, and only obscure it partially, rather than taking over the screen entirely. Freestar allows you to display either small (320x50) or medium (300x250) banners, and to choose to show them at the top, bottom, or middle of the device screen. You can also receive callbacks for key events in the ad's lifecycle.

To do this, your Unity code needs to implement the `FreestarBannerAdCallbackReceiver` interface:

```
void onBannerAdShowing(string placement, int adSize);
void onBannerAdClicked(string placement, int adSize);
void onBannerAdFailed(string placement, int adSize);
```

And then pass in the object that implements this interface to `FreestarUnityBridge`:

```
FreestarUnityBridge.setBannerAdListener(this);
```

Finally, you request the ad. There are no separate load and show calls:

```
FreestarUnityBridge.ShowBannerAd(
  "", //the placement string -- empty if you don't use multiple placements
  FreestarConstants.BANNER_AD_SIZE_320x50,  //banner size
  FreestarConstants.BANNER_AD_POSITION_TOP  //position
);
```

The ad will load and display automatically at the appropriate position. When you no longer need it, you can remove it:

```
FreestarUnityBridge.CloseBannerAd("",FreestarConstants.BANNER_AD_SIZE_320x50);
```

Make sure the placement string and size you pass in match.

Working sample

Take a look at the sample project and specifically its `Main.cs` class.

This working sample shows how to initialize, implement the ad listeners, and show fullscreen Interstitial, Rewarded, and banner ads. **SampleScene.scene** is also included.

Some publishers like to create a singleton class based on this sample. Utilize it as you see fit.

The generated project

After creating the app flow in Unity, you can have it generate an Xcode project: `File->Build Settings` will open the settings window, with a `Build` button. This will only work on a Mac that has Xcode and

Cocoapods installed; you will be prompted for a save location, and Unity will create an Xcode project. Within the generated folder, make sure to open the `.xcworkspace` file. You will now have a generated Xcode project capable of building a working iOS app with Freestar Ads integrated.

However, there are several things to keep in mind. If, in the `FreestarMediationDependencies.xml` file (see above), you've asked to include either `FreestarAds-Googleadmob` or `FreestarAds-GAM` pods, you will need to modify the generated app's `Info.plist` file to include the following:

```
<key>GADApplicationIdentifier</key>
<string>{YOUR_ADMOB_KEY}</string>
```

The key is obtained from Google, without it, an app that integrates either of its Ad SDKs cannot run.

There are also potential build errors due to Unity-iOS compatibility issues, that depend on the version of Unity and Xcode involved. Two known ones are resolved as follows:

A Metal compatibility problem

Some version of Unity generate invalid Metal-related code; this produces a compilation error when attempting to build the app in the `MetalHelper.mm` file. The issue can be resolved by commenting out the following section:

```
#if PLATFORM_IOS || PLATFORM_TVOS
   if (@available(iOS 10.3, tvOS 10.2, *))
   {
      const int targetFPS = UnityGetTargetFPS(); assert(targetFPS > 0);
      [UnityCurrentMTLCommandBuffer() presentDrawable: surface->drawable afterMinimumDuration: 1.0 / targetFPS]
;
      return;
   }
#endif
```

There is a fallback outside the macro, so the removal of the code does not break the rest of the architecture.

An Unity 2018 problem

iOS apps generated with Unity 2018 will not build for the simulator in recent versions of Xcode. Builds for device work correctly.

A module header problem

Unity Xcode projects are generated with modules disabled, this means convenience headers in the style:

`@import UIKit;`

don't work. Some versions of the FreestarAds SDK include such a declaration in the `FreestarPrerollAd.h` header. Xcode will flag the compiler error, the solution is to open the header and replace the statement:

```
@import AVKit;
```

With the equivalent:

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

Now, the Xcode will build the app and you can run it on your device or simulator as if it was a native app.

Important iOS 14 Changes

One thing you must do as a publisher is set the Facebook `setAdvertiserTrackingEnabled` flag appropriately, since Facebook is one of our partners. Here is the official document from Facebook. Generally speaking, one typical you could approach this is if the user has granted the app permission to track them at the OS level, then you would set this flag to true. By default, the flag will be false, meaning that Facebook will not serve any ads. We leave this implementation detail to you in order to give you, the app publisher, more control.

GDPR Update

We are releasing a major update to the Freestar SDK later this week that will require publishers to utilize a GDPR TCF 2.0 compliant CMP in order to render ads to users in a GDPR affected country. If the publisher is not utilizing a GDPR TCF 2.0 compliant CMP once upgraded to this SDK version, then ads will not serve to users in a GDPR affected country.

We put together a comprehensive list of FAQs below to help break down how the changes may impact your business. If you have additional questions, please reach out to your dedicated Account Manager and we will be happy to help on a case by case basis.

What is the new SDK version?

4.0.0 for iOS and Android

How do I become GDPR compliant?

In order to be able to serve ads to your users who reside in any EU country, you will need to implement an IAB TCF 2.0 CMP service.

The CMP service is essentially a personal data and privacy form that must be presented to users to collect their consent or dissent. This form can be presented as often as you like throughout the session of your game or app, as you may require your users to see ads. The most typical setup would be for the first visit to an App with an option to edit your preferences through some other

call to action.

What if I already have a CMP implemented?

If you already have a CMP implemented, you will need to add Publisher First, Inc. (Freestar's official legal entity) as a vendor, as well as each of our partners listed here if not already included.

You will need to retrigger the consent form for GDPR affected users once Freestar and additional vendors are added to the vendor list.

If the list of vendors has already been included, no additional effort is required.

How does the Freestar SDK work with a CMP?

Freestar SDK will automatically detect the user's CMP response and act accordingly. More specifically, if the user consents, then Freestar SDK will be allowed to serve ads. If the user dissents, then Freestar SDK will not show ads.

What if I choose not to implement a CMP?

If a CMP is not implemented at all, then Freestar SDK will not show ads to users who reside in a GDPR affected country.

What do I need to do once I choose a CMP?

After you have chosen a CMP Service provider, during configuration and setup of your CMP, you will need to include a list of supported vendors. We recommend that you select 'Include All Vendors' rather than choosing specific vendors, as your list of vendors may change over time.

If you choose to add vendors to your list manually, you will need to include Publisher First, Inc. (Freestar's official legal entity) as a vendor as well as Freestar's list of supported vendors here.

Does Great Britain still observe GDPR requirements despite leaving the E.U.?

Yes, it does still apply. In anticipation of Brexit, a new domestic data privacy law called the UK-GDPR took effect on 1/31/20. The UK-GDPR is almost word for word completely identical to the EU's GDPR.

CMP Recommendations:

Consent Manager - https://www.consentmanager.net/ App Consent - https://sfbx.io/en/produits/

Here is a list of IAB approved CMP Service providers you can implement in your game or app: https://iabeurope.eu/cmp-list/

Testing

For iOS, please use our iOS test key **91784edd-3492-4111-8742-f71bd3803dd3** (we have a different

key for Android, so don't use for both iOS and Android) for all your iOS testing runs and enable test mode.

Turn on test mode:

```
FSTR_TEST_ADS
```

You will usually get 100% fill on all ad units.

It is not recommended to use your production key for testing runs as that is strictly prohibited by our partners and bad things may happen to us on the business side of things.
Do not forget to uninstall and re-install your app when changing keys on your device.

When you are satisfied with your testing, please make a release build with your production key, and turn test mode off. Publish to store.

SKAdNetwork IDs

iOS 14 changed the way advertising works on iOS devices. Ad effectiveness tracking requires the usage of `SKAdNetwork` APIs. To enable this, `SKAdNetwork` keys should be included in your app's `Info.plist` file.

Please see our list: https://github.com/freestarcapital/SDK_documentation_iOS/wiki/iOS-14-SKAdNetwork-IDs

M1 Macs

Currently, we do not support the building of our SDK on M1 macs. This will be addressed in an upcoming release.

Xcode 12 related cocoapods build errors

Recently, in Xcode 12.2, Apple made some breaking changes in Xcode related to build settings. By default, arm64 architecture is now added to ARCHS_STANDARD. In addition, they have removed support for VALID_ARCHS setting from Xcode. Apple also added, in Xcode 12, a new build setting called **Excluded Architectures**. The reason Apple added arm64 is to support simulator on M1 Mac hardware. However this causes build issues running simulator on Intel Macs. As a result, publishers running Xcode on Intel Macs will need to make usage of the EXCLUDED_ARCHS build setting, for simulator testing, to exclude the arm64 simulator. So, if your Xcode build is failing with arm64 simulator errors in the log, such as:

`building for iOS Simulator, but linking in object file built for iOS, file for architecture arm64`

then, it is suggested to use this cocoapods post install hook in your Podfile:

```
post_install do |installer|
    installer.pods_project.build_configurations.each do |config|
        config.build_settings["EXCLUDED_ARCHS[sdk=iphonesimulator*]"] = "arm64"
    end
end
```

This is a temporary fix and should address the issue of build failures resulting from Xcode attempting to build (arm64) simulator on Intel Macs. As of our latest SDK release, we do this automatically in the podspec via xcconfigs, however not every SDK vendor has adopted this workaround. This post install hook is only needed if you are running Xcode 12 on Intel hardware and wanting to run simulator, and you have a cocoapods dependency that is not Xcode 12 / arm64 compliant.

Firebase Dependency

If your app integrated the Firebase SDK, reference the Firebase version compatible with current release 7.0

pod 'Firebase/Core', '~> 7.0'

For further reference:
https://github.com/CocoaPods/CocoaPods/issues/10104
https://stackoverflow.com/questions/63607158/xcode-12-building-for-ios-simulator-but-linking-in-object-file-built-for-ios/63955114#63955114

Yahoo Demand Partner Configuration

If Yahoo is being added as a demand partner in your Podfile, then it is required to add an entry into your Info.plist. See example below:

```
<key>VerizonAdsSourceAppId</key>
<string>Replace_this_with_your_App's_App_Store_ID</string>
```

GAM and Googleadmob OB adapters

If you would like to make usage of Google Open Bidding (OB) adapters for FreestarAds, please contact your account manager (AM) for further instructions. We do not advise including OB adapters into your project without first consulting with your AM.