AdLib Plugin for iOS & Android

Unity Usage

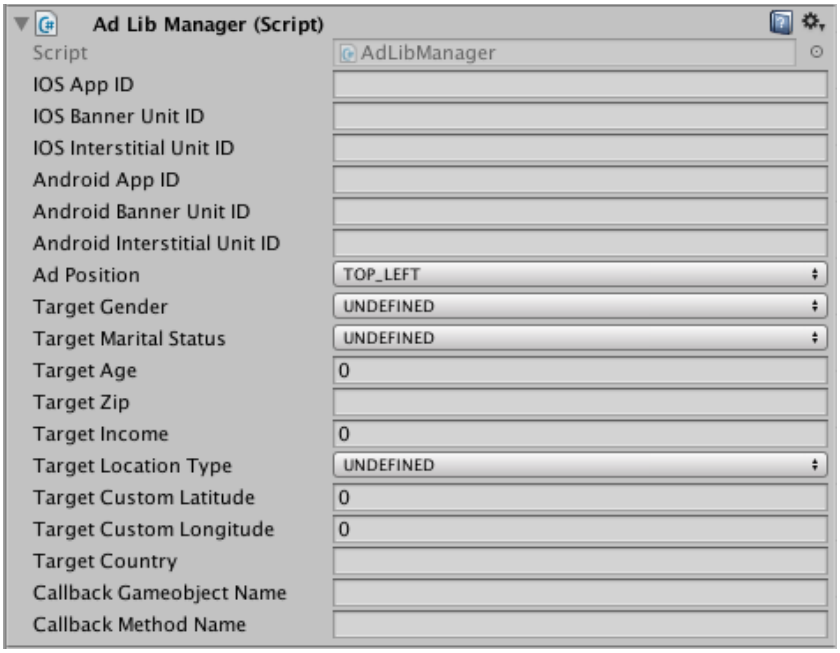
1. Import the AdLibUnityPlugin.unitypackage into your project.

2. Under Assets > Prefabs, click on the AdLibManager prefab.

3. In the Inspector, you will see various fields where you may fill in or select the appropriate information. The information is described in the table below.

Field	Type	Description
iOS App ID	string	The AdLib iOS App ID
iOS Banner Unit ID	string	The AdLib iOS Banner Unit ID
IOS Interstitial Unit ID	string	The AdLib iOS Interstitial Unit ID
Android App ID	string	The AdLib Android App ID
Android Banner Unit ID	string	The AdLib Android Banner Unit ID
Android Interstitial Unit ID	string	The AdLib Android Interstitial Unit ID
Ad Position	AdPosition	The desired position of the Banner Ad. Possible Values: TOP_LEFT, TOP_CENTER, TOP_RIGHT, CENTERED, BOTTOM_LEFT, BOTTOM_CENTER, BOTTOM_RIGHT
Target Gender	Gender	The desired target gender. Possible Values: UNDEFINED, MALE, FEMALE
Target Marital Status	MaritalStatus	The desired target marital status. Possible Values: UNDEFINED, MARRIED, SINGLE
Target Age	int	The desired target age
Target Zip	string	The desired target zip code
Target Income	int	The desired target income
Target Location Type	TargetLocationType	The type of location targeting. Select UNDEFINED if no location targeting is needed. Select USER_LOCATION if you would like to constantly target the user's current location (iOS Only). Select CUSTOM_LOCATION if you would like to enter in latitude and longitude coordinates (Android Only)
Target Custom Latitude	double	The desired target longitude if CUSTOM_LOCATION is selected for Target Location Type (Android Only)

Target Custom Longitude	double	The desired target longitude if CUSTOM_LOCATION is selected for Target Location Type (Android Only)
Target Country	string	The desired target country (iOS Only)
Callback Gameobject Name	string	The GameObject which will receive AdLib callbacks
Callback Method Name	string	The method in the script attached to the Callback Gameobject that will receive the callbacks



4. Drag and drop the AdLibManger prefab in each of the scenes where you plan to use AdLib. Make sure that all of the information needed is filled in the inspector for the AdLibManager prefab.

5. It is HIGHLY recommended to take advantage of the AdLib callbacks so that you may utilize AdLib functions in their proper order. For example, you cannot load interstitial ads without first having interstitial ads initialized. In an active script of your choice, add this function to receive callbacks. Make sure the GameObject of which the script is attached to is named exactly how you filled it in for Callback Gameobject Name in the inspector. Make sure the function name is exactly as you filled it in for Callback Method Name in the inspector. Where applicable, fill in your desired code in each of the cases:

```
public void AdLibCallback(string message)
{
	switch(message)
	{
		case "ADLIB_INITIALIZED":
		{
			break;
		}
		case "ADLIB_BANNER_INITIALIZED":
```

```
                {
                        break;
                }
                case "ADLIB_BANNER_LOADED":
                {
                        break;
                }
                case "ADLIB_BANNER_FAILED":
                {
                        break;
                }
                case "ADLIB_BANNER_TAPPED":
                {
                        break;
                }
                case "ADLIB_INTERSTITIAL_INITIALIZED":
                {
                        break;
                }
                case "ADLIB_INTERSTITIAL_LOADED":
                {
                        break;
                }
                case "ADLIB_INTERSTITIAL_FAILED":
                {
                        break;
                }
                case "ADLIB_INTERSTITIAL_SHOWN":
                {
                        break;
                }
                case "ADLIB_INTERSTITIAL_DISMISSED":
                {
                        break;
                }
        }
}
```

6. To call any method from AdLib within any of your classes, reference the AdLibManger as such, following the method you would like to call.

   AdLibManager.Instance.<Desired Method>

   For Instance, to initialize AdLib:

   AdLibManager.Instance.InitializeAdLib();

7. Refer to this list of available methods which you may call:

Method	Description
InitializeAdLib()	Initializes AdLib with the specified AdLib App ID Banner Unit ID, and Banner Position

InitializeBannerView()	Initializes the AdLib Banner View and begins loading banner ads
SetBannerPosition(AdPosition position)	Changes/Sets the position of the AdLib Banner Ad view
ShowBannerAd()	Shows the AdLib Banner ad view
RefreshBannerAd()	Refreshes the AdLib Banner Ad
HideBannerAd()	Hides the AdLib Banner Ad view
InitializeInterstitialView()	Initializes the AdLib Interstitial Ad View
LoadInterstitialAd()	Loads an interstitial ad
ShowInterstitialAd()	Shows the interstitial ad
RemoveAllAds()	Removes all ad views
DestroyInstance()	Destroys the instance of the AdLibManager

Android Integration

1. If you would like to use an existing AndroidManifest.xml or use the AdLib AndroidManifest.xml, make sure that these fields are included:

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>


<activity android:name="adlibmediation.adlibunityplugin.AdLibViewController ”
android:label="@string/app_name"/>

<activity android:name="com.applovin.adview.AppLovinInterstitialActivity"/>

<activity android:name="com.applovin.adview.AppLovinConfirmationActivity"/>

<meta-data android:name="com.google.android.gms.version"
android:value="@integer/google_play_services_version"/>

<activity android:name="com.google.android.gms.ads.AdActivity" android:configChanges="keyboard|
keyboardHidden|orientation|screenLayout|uiMode|screenSize|smallestScreenSize"
android:theme="@android:style/Theme.Translucent"/>
```

   If you are unsure if your AndroidManifest.xml is structured correctly, refer to the imported AndroidManifest.xml from the AdLib Unity Package. The manifest must be located under Assets > Plugins > Android . You project must include only **one** AndroidManifest.xml

2. Make sure that you have the appropriate fields filled in for the AdLibManager Prefab in the

inspector. At minimum, you must have the Android App ID, the Android Banner Unit ID and the Ad Position or Android Interstitial Unit ID filled in. AdLib will not initialize otherwise.

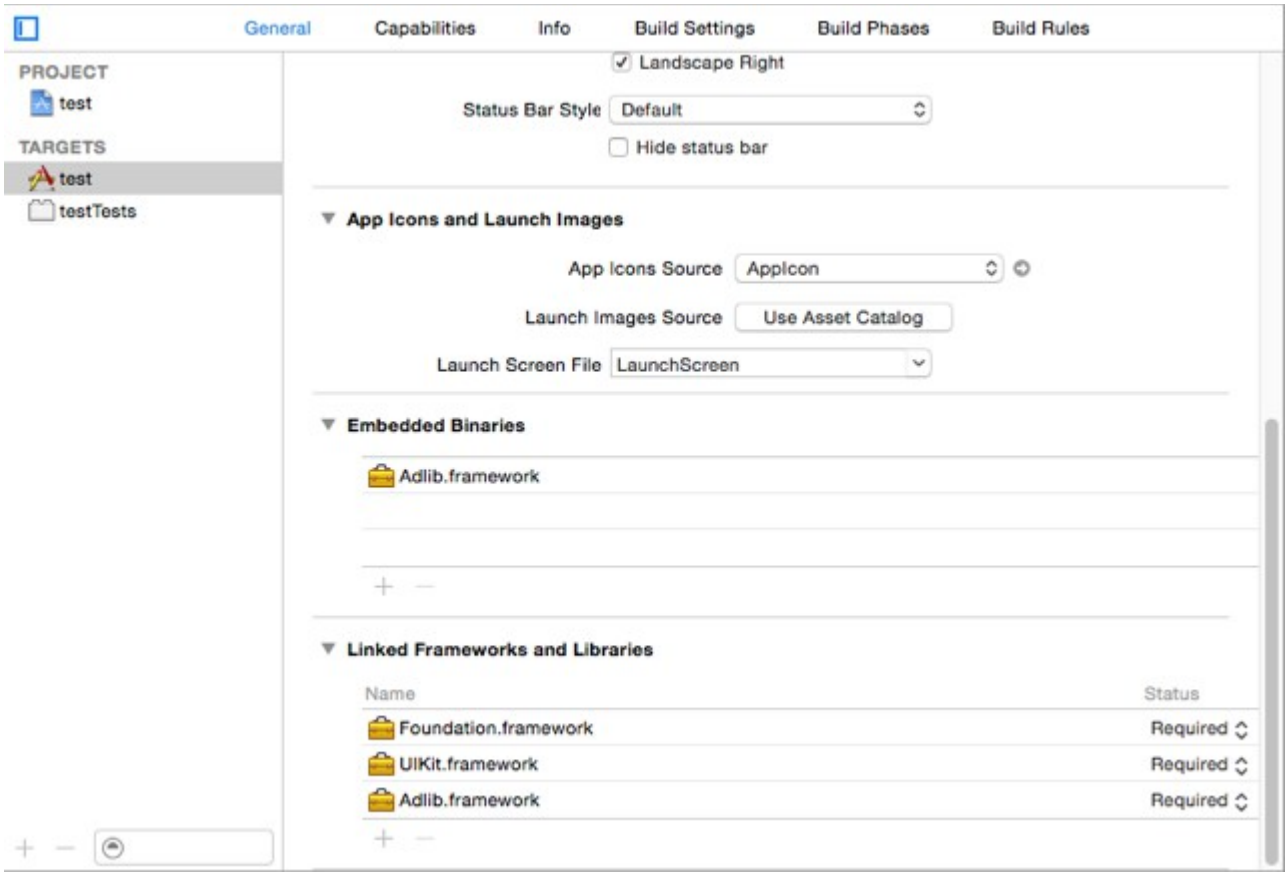
iOS Integration

1. Make sure that you have the appropriate fields filled in for the AdLibManager Prefab in the inspector. At minimum, you must have the iOS App ID, the iOS Banner Unit ID and the Ad Position or the iOS Interstitial Unit ID filled in. AdLib will not initialize otherwise.

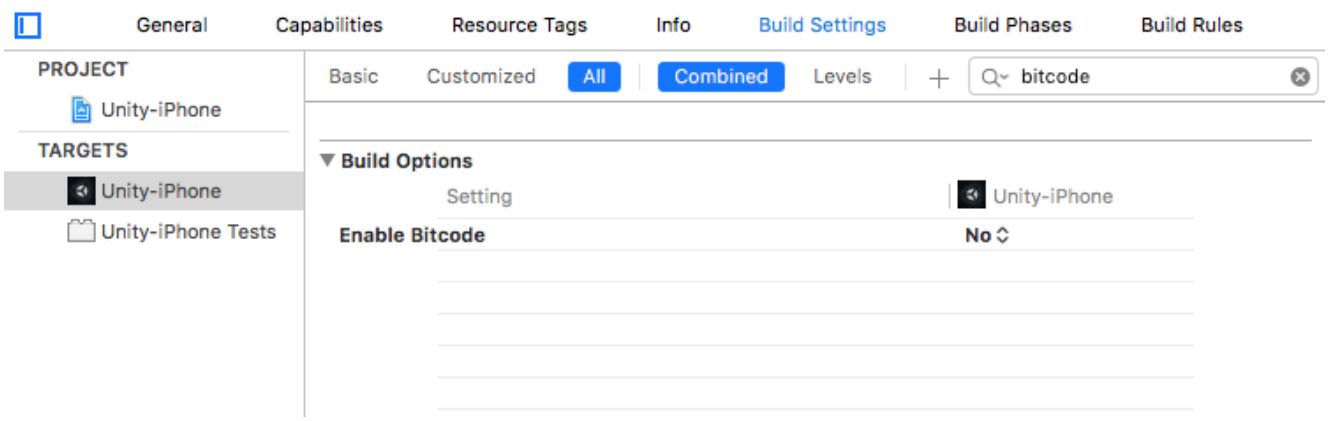
2. After building your Unity Project for iOS, open your iOS project in Xcode.

3. There are two Adlib.framework files included in the AdLib Unity Plugin download. To test your project on an iOS device and to submit your project to iTunes Connect, use the framework under iOS Device Framework file. To test your project on the simulator, use the framework under the iOS Simulator Framework file.

4. Drag and drop or manually add the according Adlib.framework file under the Embedded Binaries section in your project's target General settings.



5. Make sure Enable Bitcode is set to “No” in your build settings.


General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
PROJECT
Unity-iPhone
TARGETS
Unity-iPhone
Unity-iPhone Tests
Basic
Customized
All
Combined
Levels
bitcode
Build Options
Setting
Unity-iPhone
Enable Bitcode
No