

# Google Mobile Ad(AdMob)

Anyone can comment

---

[Short Overview](#)

[Setup](#)

[How to update](#)

[Getting Started](#)

[Before you begin](#)

[General Info](#)

[Setting up google ad with your App](#)

[Setting up an App](#)

[Setup for IOS](#)

[Requirements](#)

[Plugin Setup](#)

[Setup for Android](#)

[Requirements](#)

[Plugins Setup](#)

[API References](#)

[InAppPurchaseListener](#)

[PlayMaker Actions](#)

[Frequently Asked Questions](#)

[Example Scenes](#)

[How to get support](#)

# Short Overview

---

Plugin provides the easy and flexible functionality available in Google Mobile Ad SDK. You will be able to manage and receiving events from banners of all sizes, and interstitial ad.

Plugin Can be used with Android and IOS platforms.

Please read full documentation before using the plugin.

If you're new to IOS app development, please also read [IOS Application Setup Guide](#). You may want to read about [Compilation and signing Android Applications With Unity](#).

## Setup

All you have to do is to copy files from

**Assets/Plugins/StansAssets** to → **Assets/Plugins**

If you already have some files under **Assets/Plugins/Android** it mean that you may already have some android plugin installed. In this case, please read this sections with will help you to merge the plugins:

[Can I use this plugin with other Android Plugins from Asset Store](#)

[How to compile androidnative.jar from eclipse project](#)

[How to merge manifest with another android plugin](#)

If the plugin has conflict with the IOS / WP8 / Android (with not overrides main activity)

Please [contact support team](#).

With every new update I make try to make plugin better. Add new features, improve stability, usability and code base structure.

## Google Mobile Ads SDK

## 2. Avoiding conflicts

It will be of course described in version notes. But if you simple click update in Asset Store version, you may get duplicated or conflicted files.

**Warning:** Check the [Version Notes](#) before update. If version notes contains

**Code Refactor:** version section. It means that plugin structure is changed. Some files can be removed or function names changed. If files was removed, you should remove it by your self from the project. Example or Remove notes:

Removed:

Assets/Extensions/AndroidNative/Other/Twitter

All Scene under xExample/Scenes Moved to corresponding folder  
only Preview Scene should be under xExample/Scenes

**Note:** If you own another plugins with also have `GooglePlayCommon` folder (this folder is shared between few plugins in order to supply compatibility of android plugins) I also recommend update those plugins too. To avoid conflicts

# Getting Started

## *Before you begin*

1. Sign up as an [AdMob](#) or [DFP](#) publisher.
2. [Download](#) the SDK for your particular development platform.
3. Familiarize yourself with the [AdMob advertising network](#) or [DoubleClick For Publishers \(DFP\) mobile advertising solution](#).

## *General Info*

The Google Mobile Ads SDK allows developers to easily incorporate mobile-friendly text and image banners as well as rich, full-screen web apps known as interstitials. An ever-growing set of "calls-to-action" are supported in response to user touch including direct access to the App Store, Google Play, Windows Phone 8 Marketplace, iTunes, maps, video and the dialer. Ads can be targeted by location and demographic data. The Google Mobile Ads SDK can be used by the following publisher types:

### **AdMob publishers**

Access the Google AdMob network to easily monetize your application.

### **DoubleClick For Publishers (DFP) users**

Leverage DFP to traffic, target, and serve directly-sold ads. The SDK is available to upgraded DFP ([www.google.com/dfp](http://www.google.com/dfp)) users for [Android](#) and [iOS](#) platforms.

### **AdSense publishers**

Monetize your search results pages with Google search ads.

## Setting up google ad with your App

Once you are registered, you can login to the your admob account.

### 1) Add app for monetization

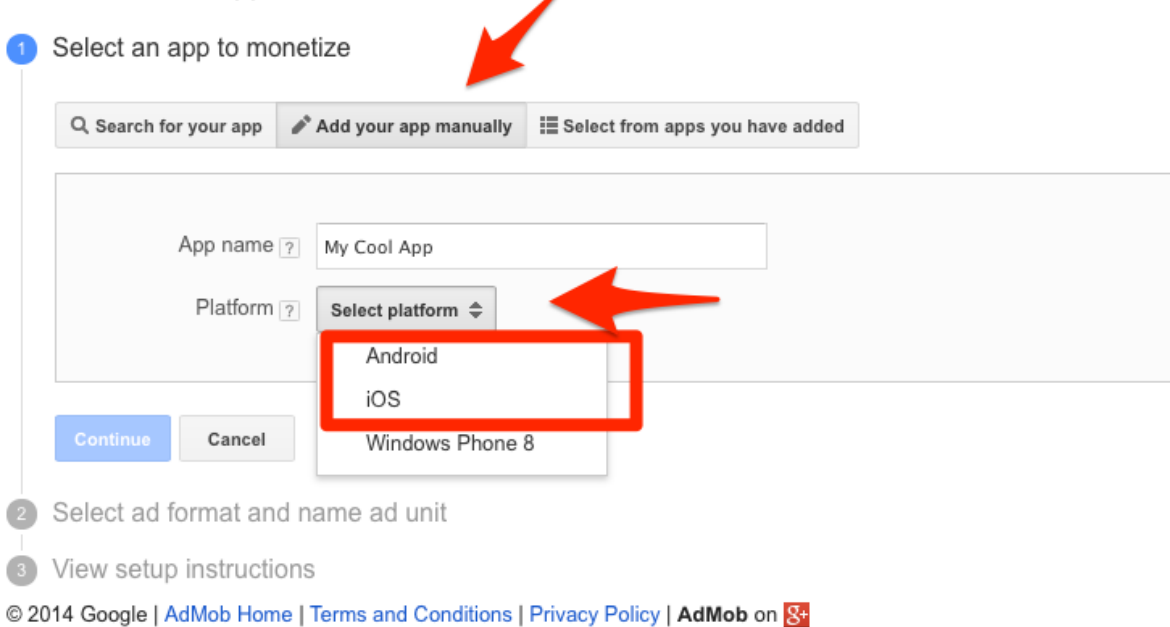


The screenshot shows the AdMob dashboard. At the top, there's a yellow banner for importing legacy AdMob data. Below that, on the left, is a section for 'Estimated earnings' with a table showing data for Today, Yesterday, This month, and Last month. On the right, there's a 'Promote your app' section. At the bottom left, a red button labeled '+ Monetize new app' is highlighted with a red arrow.

### 2) Create the app for monetization:

- Set name for your app (name will not be used in code. so you can specify any name you like).
- Choose Platform, currently plugins support Android and IOS platform.

### Monetize a new app



The screenshot shows the 'Monetize a new app' dialog box. It has three steps: 1. Select an app to monetize, 2. Select ad format and name ad unit, and 3. View setup instructions. In step 1, there are three tabs: 'Search for your app', 'Add your app manually' (which is selected and highlighted with a red arrow), and 'Select from apps you have added'. Below the tabs, there's a form with 'App name' (containing 'My Cool App') and 'Platform' (a dropdown menu). The dropdown menu is open, showing 'Android', 'iOS', and 'Windows Phone 8'. 'Android' and 'iOS' are highlighted with a red box and a red arrow. At the bottom left, there are 'Continue' and 'Cancel' buttons. At the bottom, there's a copyright notice for 2014 Google and links to AdMob Home, Terms and Conditions, Privacy Policy, and AdMob on Google+.

3) Provide the following details for the ad unit:

- **Ad unit name:** Enter a unique name and description that will help you find this ad unit later (e.g., Top Banner on Home).
- **Text ad style:** Select a text ad style that complements your app. You can use the standard style or customize your own style.
- **Automatic refresh** Determine how often a new ad impression is generated. You can choose not to refresh or to refresh ads every 30 to 120 seconds. We recommend a refresh rate between 45 and 60 seconds.
- **Google ads:** Select whether or not you'd like to use keyword targeted Google ads and Google certified ad networks to improve your app's fill rate.
- **Click Save.** You'll see the ad unit ID for this ad unit.

### Monetize a new app

✓ Select an app to monetize



[Select a different app](#)

2 Select ad format and name ad unit

Banner

Interstitial

**i** Ad type, size, and placement are specified when you integrate the code using the AdMob SDK.

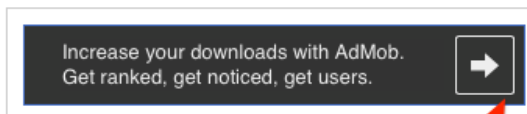
Automatic refresh ?

☐ No refresh

☒ Refresh rate: 60 seconds (30-120 seconds)

Text ad style ?

Standard



Ad unit name ?

My Banner Unity Id Name

Example: "Top Banner on Home"



4) Copy your **Ad unit id** and click Done. Ad unity Id will be used for displaying the ad in your application.

### Monetize a new app

✓ Select an app to monetize



✓ Select ad format and name ad unit

Ad unit ID: **ca-app-pub-6101605888755494/9934310764**

Ad unit name: **My Banner Unity Id Name**

3 View setup instructions

#### Set up AdMob ad units

Follow the Google Developers website for [complete instructions](#) on how to integrate the [Google AdMob SDK](#).

 Send an email with these instructions

# Setup for iOS

## Requirements

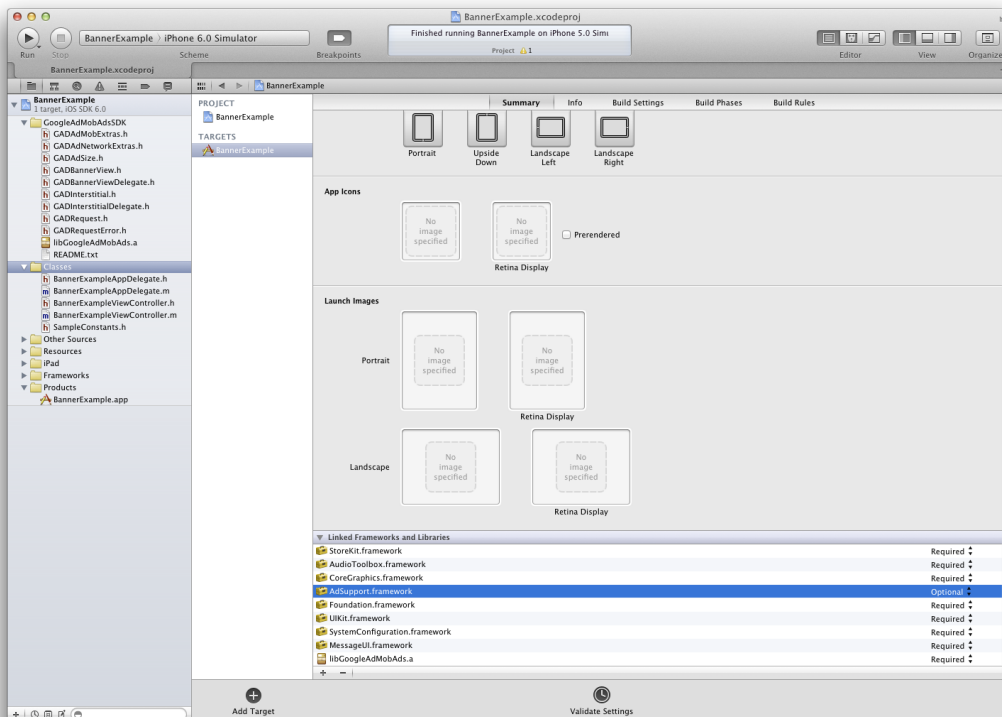
- iOS version 4.3 or later
- Xcode 4.5 or later

## Plugin Setup

1. The SDK library references the following iOS development frameworks which may not already be part of your project:

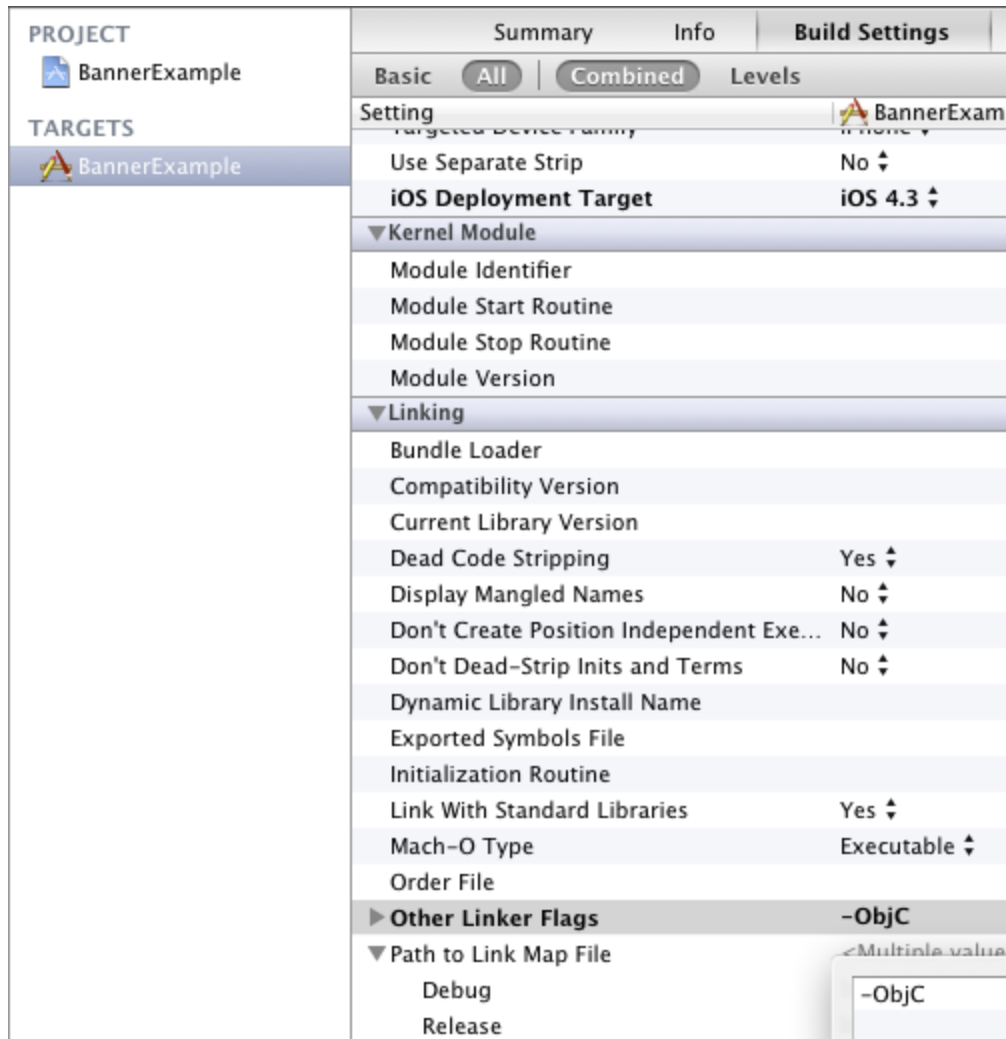
- CoreTelephony
- MessageUI
- StoreKit
- AdSupport

To add these frameworks, double-click the your project name. **Open the Link Binary With Libraries** dropdown under the **Build Phases** tab. Add the frameworks from the iOS SDK using the + button that becomes visible.



2. You now need to add **-ObjC** to the **Other Linker Flags** of your application target's build setting:

1. In Xcode's project navigator, press the blue top-level project icon.
2. Click on your target, then the **Build Settings** tab.
3. Under **Other Linker Flags**, add **-ObjC** to both **Debug** and **Release**.



You should now be able to rebuild your project without any errors. And start using plugin functions.

# Setup for Android

## Requirements

- Make sure you have the [latest copy of the Android SDK](#) and that you're compiling against at least Android v3.2 (set *target* in *project.properties* to *android-13*).
- The Google Mobile Ads SDK for Android requires a run-time of Android 2.3 or later (set *android:minSdkVersion* to at least 9 in your *AndroidManifest.xml*). This means you can develop with the latest version of the Android SDK and your app will still run on an earlier Android version (2.3 minimum).

## Plugins Setup

Make sure that [androidnative.jar](#) and [AndroidManifest.xml](#) is inside your **Assets/Plugins/Android** folder.

Open [AndroidManifest.xml](#) and enter your package name in the *package* attribute of the `<manifest>` element.

`package="REPLACE_WITH_YOUR_PACKAGE_ID"` - replace this line with your Package ID

# API References

GoogleMobileAd : Singleton<GoogleMobileAd> class.

---

## API methods:

*Init ad with your ad unit id. This function should be called before any other function of this class.*

```
public static void Init(string ios_unit_id, string android_unit_id)
```

*Changes ad unity id for banners*

```
public static void SetBannersUnitID(string ad_unit_id)
```

*Changes ad unity id for interstitials*

```
public static void SetInterstitialsUnitID(string ad_unit_id)
```

*Creates banner ad using [TextAnchor](#)*

```
public static GoogleMobileAdBanner CreateAdBanner(TextAnchor anchor, GADBannerSize size)
```

*Creates banner ad using custom x/y position*

```
public static GoogleMobileAdBanner CreateAdBanner(int x, int y, GADBannerSize size)
```

*Destroy banner by id*

```
public static void DestroyBanner(int id)
```

*Add keyword for targeting purposes*

```
public static void AddKeyword(string keyword)
```

*Sets the user's birthday for targeting purposes.*

```
public static void SetBirthday(int year, AndroidMonth month, int day);
```

*Set gender for targeting purposes, use GADGenger*

```
public static void SetGender(GoogleGenger gender)
```

*This method allows you to specify whether you would like your app to be*

*treated as child-directed for purposes of the Children's Online Privacy Protection Act (COPPA) -*  
<http://business.ftc.gov/privacy-and-security/childrens-privacy>.

*If you set this method to true, you will indicate that your app should be treated as child-directed for purposes of the Children's Online Privacy Protection Act (COPPA).*

*If you set this method to false, you will indicate that your app should not be treated as child-directed for purposes of the Children's Online Privacy Protection Act (COPPA).*

*If you do not set this method, ad requests will include no indication of how you would like your app treated with respect to COPPA.*

*By setting this method, you certify that this notification is accurate and you are authorized to act on behalf of the owner of the app. You understand that abuse of this setting may result in termination of your Google account.*

*Note: it may take some time for this designation to be fully implemented in applicable Google services.*

```
public static void TagForChildDirectedTreatment(bool val);
```

*Causes a device to receive test ads. The deviceId can be obtained by viewing the logcat output after creating a new ad.*

*The device ID that AdMob accepts is a hashed value (I'm not sure, but it might also include a salt) of your actual device ID. The way to get this hashed device ID is to make a live request on your device and check logcat for a message like "To get ads on this device, call adRequest.addTestDevice("YOUR\_HASHED\_DEVICE\_ID")". This ID is what you should use. It should be a 32-digit HEX number like the numbers the OP has.*

```
public static void AddTestDevice(string deviceId)
public static void AddTestDevices(params string[] ids)
```

*Function will start interstitials banner request and will show it as soon as banner loaded.*

```
public static void StartInterstitialAd()
```

*Function will send interstitials banner request.*

```
public static void LoadInterstitialAd()
```

*Shows interstitial banner if it was previously Loaded*

`public static void ShowInterstitialAd()`

*Record IAP resolutions. [Read More](#).*

`public static void RecordInAppResolution(GADInAppResolution resolution)`

*Called when interstitial an ad is received*

`ON_INTERSTITIAL_AD_LOADED`

*Called when interstitial an ad request failed*

`ON_INTERSTITIAL_AD_FAILED_LOADING`

*Called when interstitial an ad opens an overlay that covers the screen.*

`ON_INTERSTITIAL_AD_OPENED`

*Called when the user is about to return to the application after clicking on an ad.*

`ON_INTERSTITIAL_AD_CLOSED`

*Called when an ad interstitial leaves the application (e.g., to go to the browser).*

`ON_INTERSTITIAL_AD_LEFT_APPLICATION`

*Called when ad action triggers in-app request. [Read More](#)*

`ON_AD_IN_APP_REQUEST`

**Warning:** GoogleMobileAd not event dispatcher by it self. To be able to listen for the events sign on `public static` GoogleMobileAdInterface controller getter events of GoogleMobileAd class. Controller will be created after GoogleMobileAd `init` function.

## GoogleMobileAdBanner interface.

---

### API methods:

*Hide ad banner*

`public void HideAd()`

Show ad banner (only if it was hidden by **HideAd** function)

```
public void ShowAd()
```

Refresh ad content (will send new request to google)

```
public void Refresh()
```

Change Banner Position without destroying the banner

```
public void SetBannerPosition(int x, int y)
```

```
public void SetBannerPosition(TextAnchor anchor)
```

## Get / Set:

Banner id

```
int id {get;}
```

Banner width

```
int width {get;}
```

Banner height

```
int height {get;}
```

true if banner was Loaded

```
bool IsLoaded {get;}
```

true if banner currently on screen

```
bool IsOnScreen {get;}
```

Defines show or not banner when it's Loaded.

```
bool ShowOnLoad{get; set;}
```

## Events:

Called when an ad is received

```
ON_BANNER_AD_LOADED
```

Called when an ad request failed

```
ON_BANNER_AD_FAILED_LOADING
```

*Called when an ad opens an overlay that covers the screen.*

**ON\_BANNER\_AD\_OPENED**

*Called when the user is about to return to the application after clicking on an ad.*

**ON\_BANNER\_AD\_CLOSED**

*Called when an ad leaves the application (e.g., to go to the browser).*

**ON\_BANNER\_AD\_LEFT\_APPLICATION**

# InAppPurchaseListener

Note: You will only receive in-app purchase (IAP) ads if you specifically configure an IAP ad campaign in the AdMob front end.

Implement the `onInAppPurchase` listener, is really easy, all you have to do is to subscribe to `ON_AD_IN_APP_REQUEST` event

```
GoogleMobileAd.addEventListener(GoogleMobileAdEvents.ON_AD_IN_APP_REQUEST,  
OnInAppRequest);
```

Implement Event data will contain product id. You should start your game purchase flow with this id as soon as you will receive `ON_AD_IN_APP_REQUEST` event

```
private void OnInAppRequest(CEvent e) {  
    //getting product id  
    string productId = (string) e.data;  
    Debug.Log ("In App Request for product Id: " + productId + " received");  
    //Start purchase flow with productId here  
}
```

Once the purchase is complete, you should call `RecordInAppResolution` with one of the following constants defined in `GADInAppResolution`:

```
public enum GADInAppResolution {  
    RESOLUTION_SUCCESS = 0,          // Purchase was a success  
    RESOLUTION_FAILURE = 1,          // Error while processing purchase  
    RESOLUTION_INVALID_PRODUCT = 2,  // Error while looking up product  
    RESOLUTION_CANCELLED = 3         // Purchase was cancelled by user  
}
```

An example of a success call would look like this:

```
GoogleMobileAd.RecordInAppResolution(GADInAppResolution.RESOLUTION_SUCCESS);
```

# *PlayMaker Actions*

The plugin now contains playmaker actions.

The actions scripts can be found in the zip archive at:

**Assets/Extensions/GoogleMobileAd/Addons/PlayMakerActions**

You can simply unrar it to the same folder and Google Mobile Ad actions will appear under playmaker actions menu. You always welcome on the [PlayMaker Actions Forum Thread](#) to request new actions or report a bug

The current action list is:

GAD\_InitGoogleAd

GAD\_SetAdTargeting

GAD\_SetAdTestDevices

GAD\_CreateBanner

GAD\_ShowBanner

GAD\_HideBanner

GAD\_RefreshBanner

GAD\_DestroyBanner

GAD\_StartInterstitialAd

GAD\_LoadInterstitialAd

GAD\_ShowInterstitialAd

# Frequently Asked Questions

---

***I have one unit id for banners ad and one for interstitial, but init function is taking only one id. How should I specify both of them?***

If you have two ids to specify, you can init admob controller with for example banners id, and then set id specifically for interstitial

***How do I get an AdMob ad unit ID?***

Directions for how to create an AdMob ad unit ID can be found [here](#). AdMob ad unit IDs have the form **ca-app-pub-XXXXXXXXXXXXXXXXX/NNNNNNNNNN**.

***I keep getting the error 'The Google Play services resources were not found. Check your project configuration to ensure that the resources are included.'***

You can safely ignore this message. Your app will still fetch and serve banner ads.

***I keep getting the error 'Invalid unknown request error: Cannot determine request type. Is your ad unit id correct?'***

Make sure your ad unit ID is correct. For publishers using the new AdMob front end, the ad unit ID will be in the form **ca-app-pub-XXXXXXXXXXXXXXXXX/NNNNNNNNNN**.

You will get this error if you use the form **pub-XXXXXXXXXXXXXXXXX**.

***My app support autorotation. But ad banner is not changing when the app is rotated.***

The auto rotation for banners is not supported. But you can implement it by your self, I will describe algorithm below.

Most of game apps have one orientation, and those who support both as usual use different banner position and size for different orientation. that why I do not see the reason to

implement automation auto rotation.

1) 95% if user will not be use this

2) Users who will need this feature would love to add extra enchantments

So I decided to give you full control on banners instead of implementing features that you will not use.

Here is algorithm how you can use to implement custom auto rotate banner to your app.

1) App started at landscape.

2) Create banner and assign it to **LandscapeBanner** variable

3) Detected rotation to portrait

4) hide **LandscapeBanner**

5) Create new banner and assign it to **PortraitBanner** variable

6) Detected rotation to back to landscape

7) check if **LandscapeBanner** was created, if no see the step 2.

8) Hide **PortraitBanner**. Show **LandscapeBanner**

# Example Scenes

---

## Example

This example scene can be found at Assets/Extensions/GoogleMobileAd/xExample/Scenes/Example. It will describe how you can use Google Mobile Ad API. The controller script GoogleAdsExample is attached to the `_Controller` gameobject and provides example for API calls.

### Example[Setup]

The scene should work out of the box. But it uses my ad identification. In order to see ad from your account you need to complete your ad [account set up](#) and replace ad unit id's in the **GoogleAdsExample**

```
private const string MY_BANNERS_AD_UNIT_ID = "ca-app-pub-6101605888755494/1824764765";  
private const string MY_INTERSTITIALS_AD_UNIT_ID = "ca-app-pub-6101605888755494/3301497967";
```

---

## NoCodingExample

This example scene can be found at Assets/Extensions/GoogleMobileAd/xExample/Scenes/NoCodingExample. It will describe how you can use Google Mobile Ad API **without any actions**. You have two controller scripts AndroidAdMobBanner and AndroidAdMobBannerInterstitial or IOSBanners Unity Id(set size) and IOSInterstitial Unity Id are attached to the *MyBanner* and *MyInterstitial* gameobjects and provides examples for API calls.

### NoCodingExample[Setup]

The scene should work out of the box. For Banners - **AndroidBannersUnityId**(your\_ad\_unit\_id), **Size** of your banner, **Anchor**, **Width**, **Height**. or **IOSBanners Unity Id**(set size). For Interstitial - **InterstitialUnityId**(your\_ad\_unit\_id) or **IOSInterstitial Unity Id**.

---