AdMob Adapter Installation

Android Admob Adapter Installation

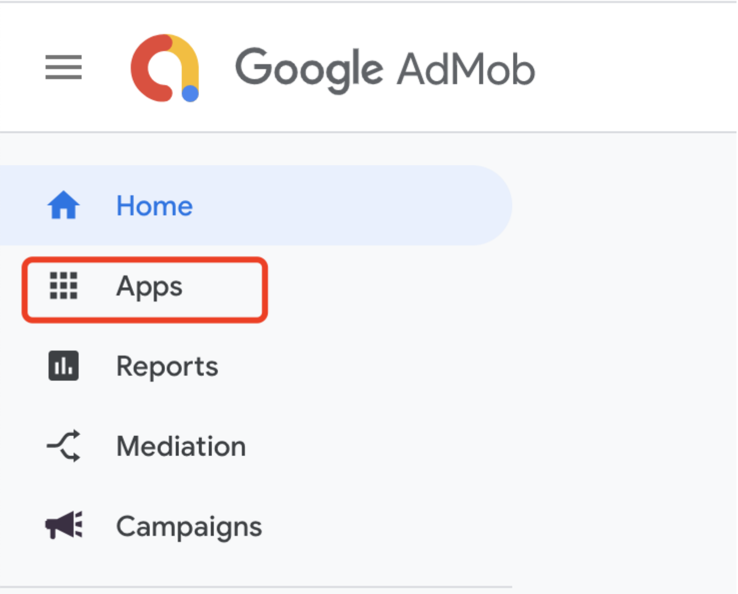
Adding AdView to AdMob

Step 1: Login to your AdMob account and configure AdView

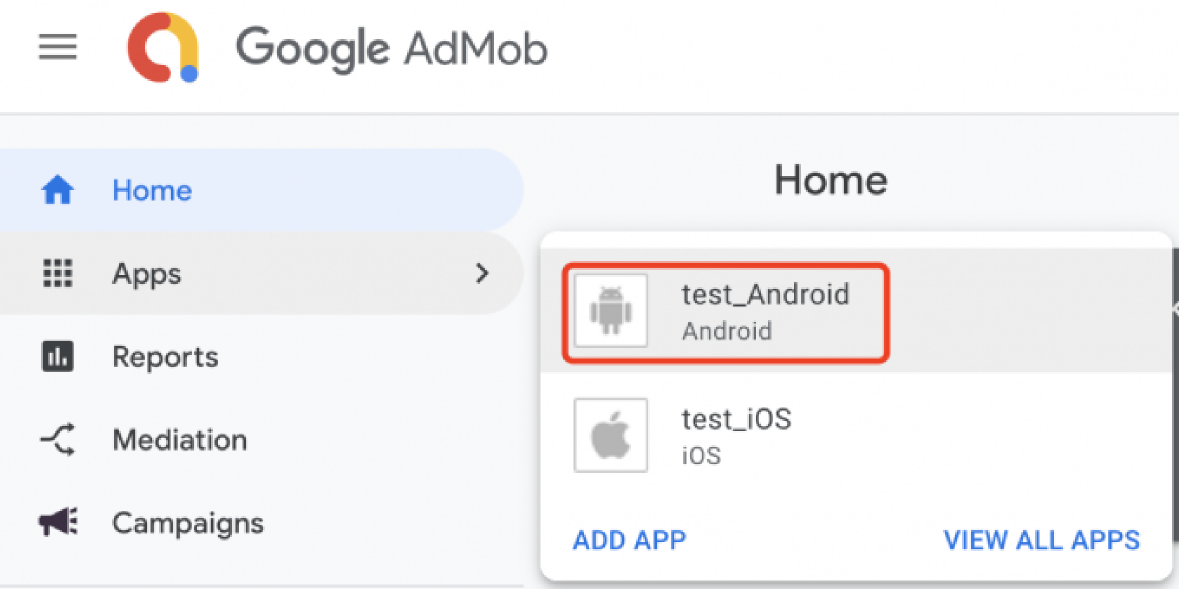
As explained above, you must have a AdMob account, and an associated app within this account, in order to integrate AdView.

Step 2: Select your app and click to monetize it

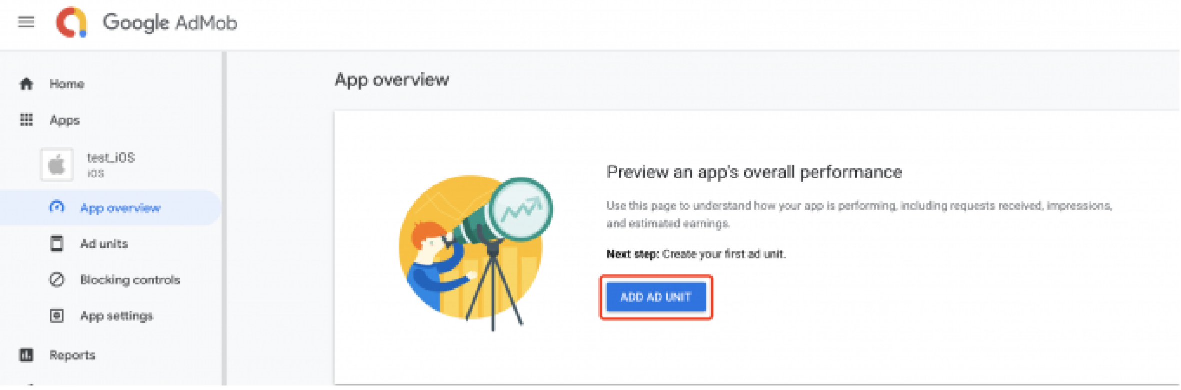
Within your AdMob account, press the 'Apps' tab



AdView must be integrated as a custom event. If your app does not have a ad unit yet, you should create a new ad unit by clicking on your app's name:



After clicking on your app's name, you will need to click the "New ad unit" button

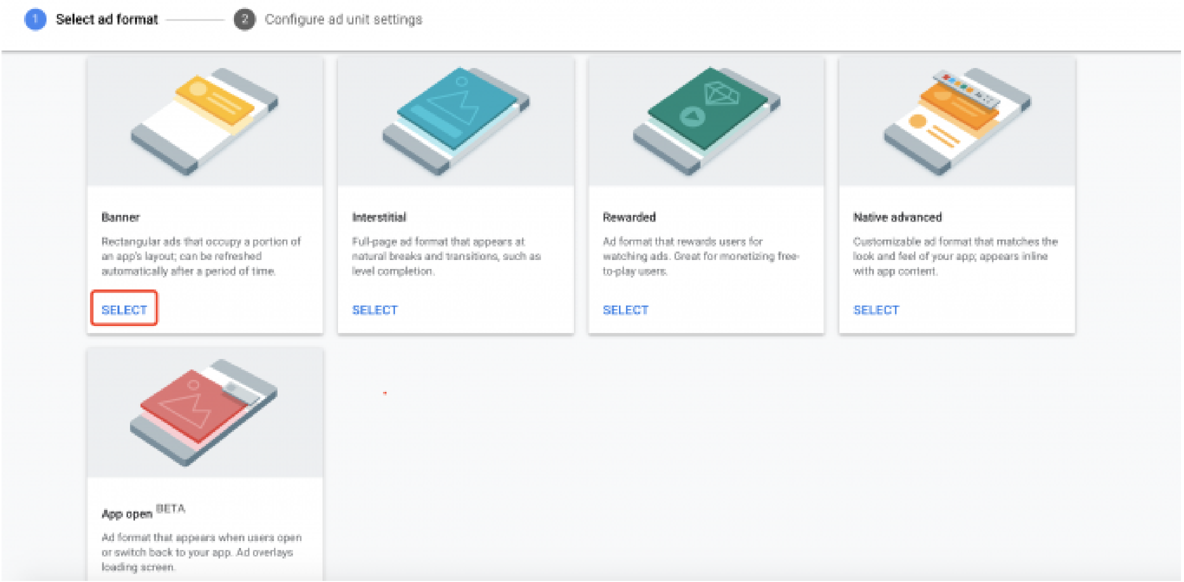


Preview an app's overall performance

Use this page to understand how your app is performing, including requests received, impressions, and estimated earnings.

Next step: Create your first ad unit.

Provide a name for your new ad unit. After making the device type selection, you should choose a format that you need to installation:



Banner

Rectangular ads that occupy a portion of an app's layout; can be refreshed automatically after a period of time.

Interstitial

Full-page ad format that appears at natural breaks and transitions, such as level completion.

Rewarded

Ad format that rewards users for watching ads. Great for monetizing free-to-play users.

Native advanced

Customizable ad format that matches the look and feel of your app; appears inline with app content.

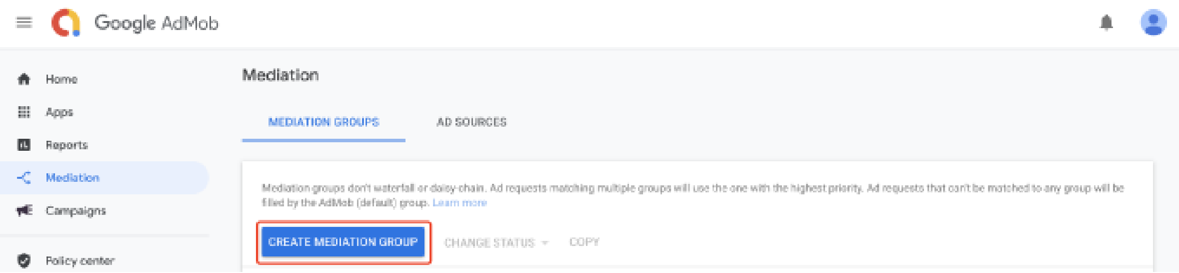
App open

Ad format that appears when users open or switch back to your app. Ad overlays loading screen.

note: create MREC AD unit, you need select "Banner". The configuration is the same as Banner.

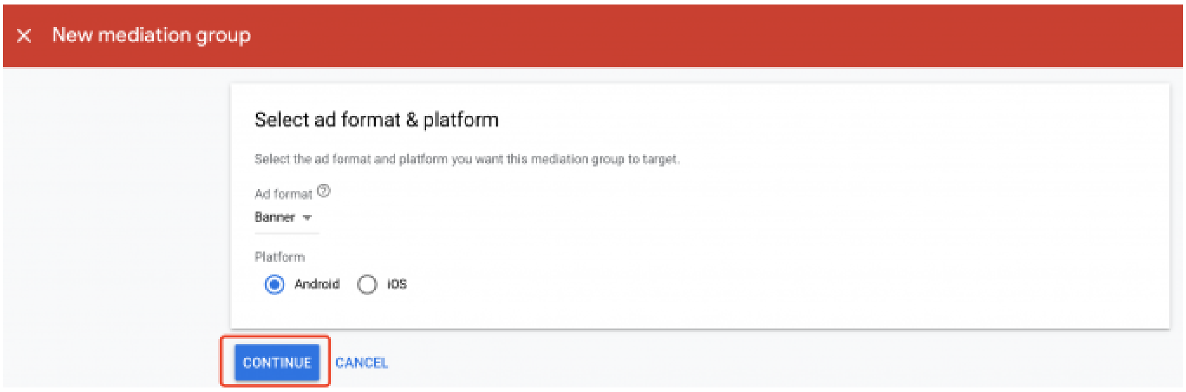
Step 3: Create mediation group for your ad placement

Within your AdMob account, press the ‘Mediation’ tab,and click "Create Mediation group".



Mediation groups don't waterfall or daisy-chain. Ad requests matching multiple groups will use the one with the highest priority. Ad requests that can't be matched to any group will be filled by the AdMob (default) group. Learn more

Then click "CONTINUE"

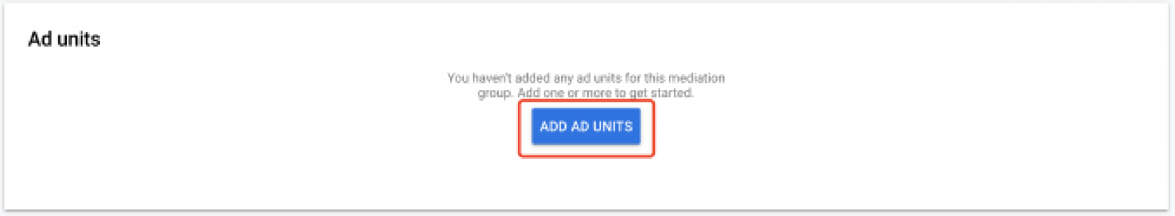


Select the ad format and platform you want this mediation group to target.

Ad format: Banner

Platform: ● Android ○ iOS

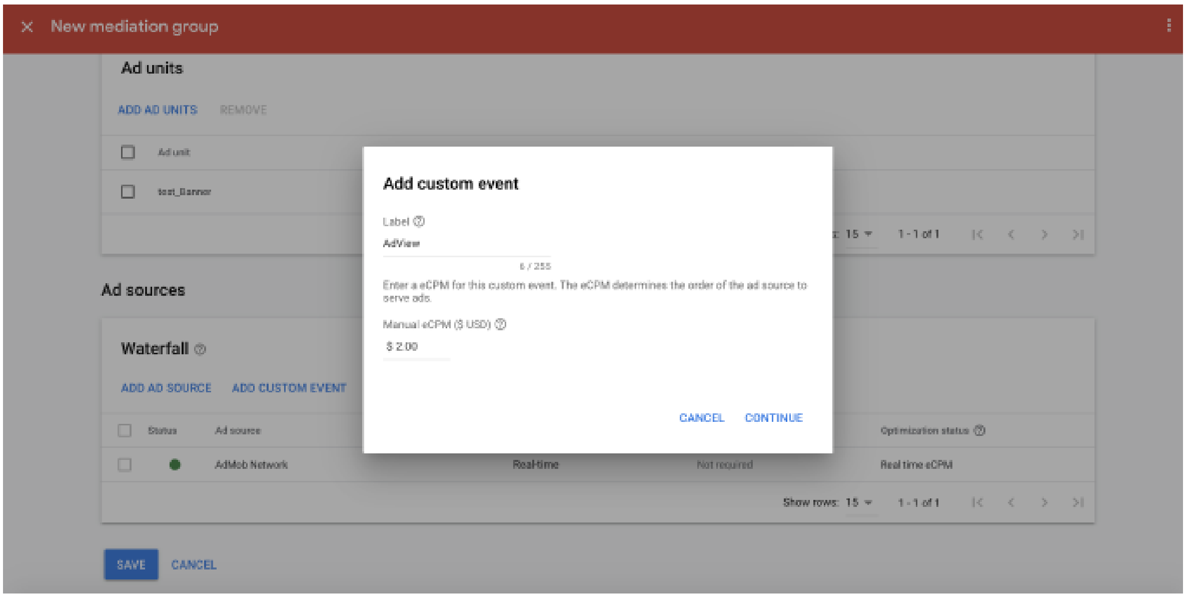
At this time, please provide a Mediation group name for your new Mediation group, then you need click"ADD AD UNITS"



Ad units

You haven't added any ad units for this mediation group. Add one or more to get started.

AdView must be integrated as a custom event,so please click Add Custom Event.When you finish write Label and eCPM click continue



Add custom event

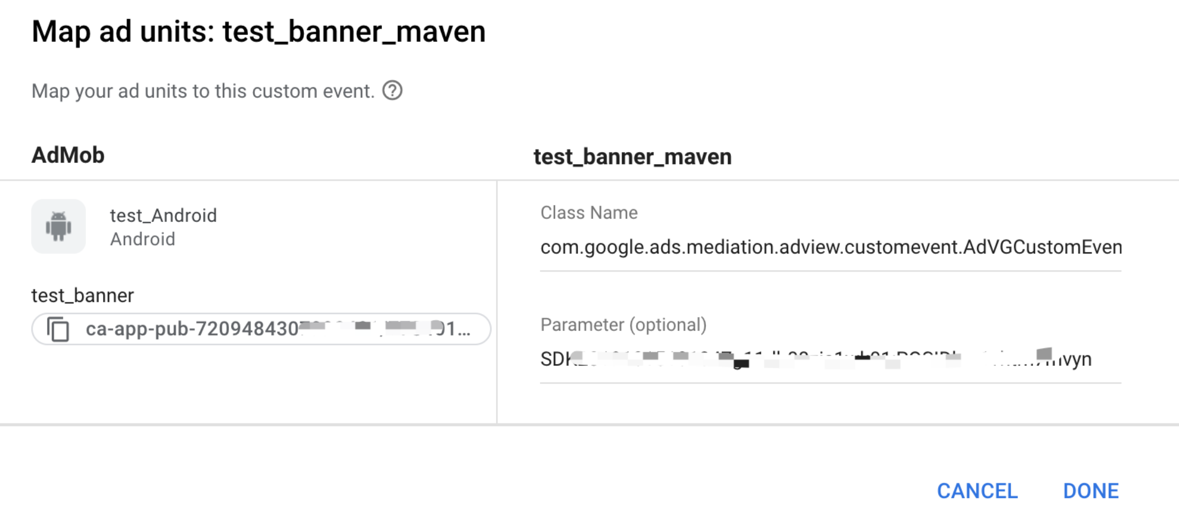
Label: AdView

6 / 255

Enter a eCPM for this custom event. The eCPM determines the order of the ad source to serve ads.

Manual eCPM ($ USD): $ 2.00

Add the following class to the "Class name" section for your ad unit



Map ad units: test_banner_maven

Map your ad units to this custom event.

AdMob	test_banner_maven
test_Android Android	Class Name com.google.ads.mediation.adview.customevent.AdVGCustomEven
test_banner ca-app-pub-720948430[redacted]1…	Parameter (optional) SD[redacted]vyn

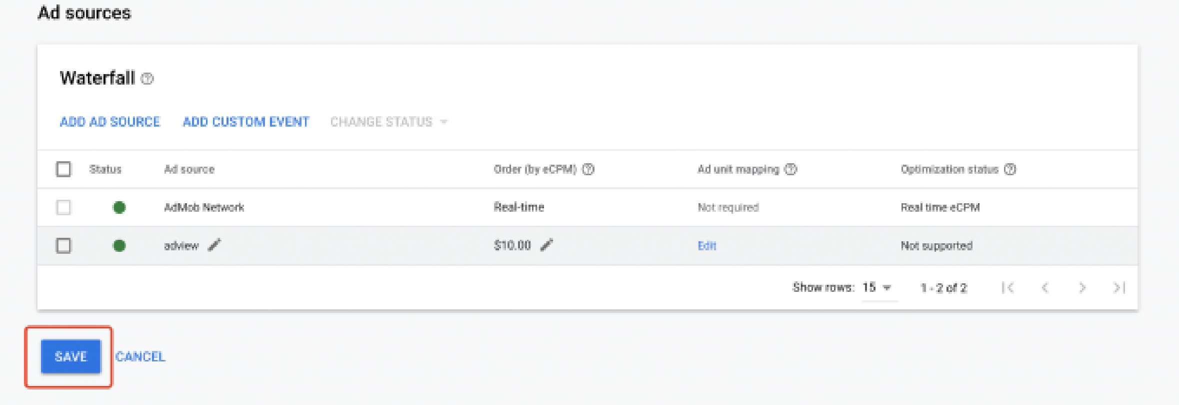
Class Name

```
com.google.ads.mediation.adview.customevent.AdVGCustomEvent
```

It is mandatory to add your AdView appId (SDK-KEY) and placement ID (posId) to the "Parameter(optional)" section.

```
AdView appId (SDK-KEY);AdView placement ID (posId)
```

Now click 'DONE' and 'SAVE'



☐	Status	Ad source	Order (by eCPM)	Ad unit mapping	Optimization status
☐	●	AdMob Network	Real-time	Not required	Real time eCPM
☐	●	adview	$10.00	Edit	Not supported

Show rows: 15 1 - 2 of 2

Now you have completed the setup, and only need simple integration to display AdView ads.

AdMob Adapter Installation

Android AdMob Adapter Installation

Step 1: Get the file with the adapter

When you see this document, you should have obtained the following documents:

- AdViewSDK_AdMobAdapter
- AdViewSDK_AdMob_Mediation_demo

Otherwise please contact your AM or partner@adview.com.

Step 2: Add the AdView SDK into your project

You can have 2 options to handle adview sdk workful.

Option 1: Pulling the latest custom adapter via mavenCentral

If you are using gradle to build your Android applications, you can pull the latest version of the SDK from mavenCentral as described below:

1. Include mavenCentral in your top-level `build.gradle` file:

```
allprojects {
    repositories {
        mavenCentral()
    }
}
```

2. Add the following line to the dependencies element in app module’s `build.gradle`.

```
api 'com.adview:android-admob-adapter:4.4.1'
```

Option 2: Adding the custom adapter Library to app module

add adview's sdk adapter lib in app path.

```
dependencies {
    api fileTree(include: ['*.*'], dir: 'libs')
}
```

Sync your gradle project to ensure that the dependency is handled by the build system.

Step 3: Use AdMob sdk in Application framework

You only need use method same as AdsMob sdk handling the customevent for AdView sdk.

here are simple example codes in app (more details pls see demo):

1.Banner

You need set adunit id and set adlistener to handle the ad events.

Note: adsize is *AdSize.BANNER*.

```
Adview adView = new AdView(view.getContext());
adView.setAdSize(AdSize.BANNER);
adView.setAdUnitId(getBannerAdUnitId());
adView.setAdListener(new AdListener() {
     @Override
     public void onAdFailedToLoad(@NonNull LoadAdError loadAdError) {
        ...
     }
});
adView.loadAd(new AdRequest.Builder().build());
```

2.Mrec

Mrec is a special banner, so just need set the adview size with banner usage.

Note: adsize is *AdSize.MEDIUM_RECTANGLE*.

```
Adview adMrecView = new AdView(view.getContext());
adMrecView.setAdSize(AdSize.MEDIUM_RECTANGLE);
adMrecView.setAdUnitId(getMrecAdUnitId()); //use mrec id
...
adView.loadAd(new AdRequest.Builder().build());
```

3.Interstitial

For interstitial ad, need load and show ad 2 steps.

```
private InterstitialAd interstitial;
```

3.1 load ad

demo codes, load ad and set InterstitialAdLoadCallback() .

```
InterstitialAd.load(MainActivity.this, getInterstitialAdUnitId(),
        new AdRequest.Builder().build(), new InterstitialAdLoadCallback() {
            @Override
            public void onAdLoaded(@NonNull InterstitialAd interstitialAd) {
                interstitial = interstitialAd;
                interstitial.setFullScreenContentCallback(
                new FullScreenContentCallback() {
                     ...
                });
                ...
             }
            @Override
            public void onAdFailedToLoad(@NonNull LoadAdError loadAdError) {
                interstitial = null;
                ...
            }
        });
```

3.2 show ad

after ad is loaded. you can show it anytime.

```
if (interstitial != null) {
        interstitial.show(MainActivity.this);
}
```

4.Reward

Reward ads also need 2 steps: load and show.

```
private RewardedAd rewardedAd;
```

4.1 load reward

load ad and set RewardedAdLoadCallback() to handle events.

```
RewardedAd.load(MainActivity.this, getRewardedAdUnitId(),new
AdRequest.Builder().build(),
      new RewardedAdLoadCallback() {
          @Override
          public void onAdLoaded(@NonNull RewardedAd ad) {
             rewardedAd = ad;
             rewardedAd.setFullScreenContentCallback(new
FullScreenContentCallback() {
                    @Override
                    public void onAdFailedToShowFullScreenContent(@NonNull
AdError error)                     {
                        ...
                    }
                    @Override
                    public void onAdDismissedFullScreenContent() {
                        ...
                    }});
             }
          @Override
          public void onAdFailedToLoad(@NonNull LoadAdError loadAdError) {
                rewardedAd = null;
                ...
            }
        });
```

4.2 show reward

Piece of demo codes.

```
if (rewardedAd != null) {
      rewardedAd.show(MainActivity.this, new OnUserEarnedRewardListener() {
          @Override
          public void onUserEarnedReward(@NonNull RewardItem rewardItem) {
                ...
          }
      });
}
```

5.Native

Native ad, need a NativeAdView to hold NativeAd. after ad has been loaded. native ad view should be added in Container.

5.1 loader and listener

```
// The ad loader.
private AdLoader adLoader;
```

demo codes, here in onNativeAdLoaded() , we put the NativeAdView into nativeContainer to show it.

```
adLoader = new AdLoader.Builder(view.getContext(), getNativeAdUnitId())

            .forNativeAd(new NativeAd.OnNativeAdLoadedListener() {
              @Override
              public void onNativeAdLoaded(@NonNull NativeAd nativeAd) {
                FrameLayout nativeContainer =
findViewById(R.id.native_container);
                NativeAdView adView = (NativeAdView) getLayoutInflater()
                    .inflate(R.layout.native_ad, null);
                populateNativeAdView(nativeAd, adView);
                nativeContainer.removeAllViews();
                nativeContainer.addView(adView);
              }
            })
            .withAdListener(new AdListener() {
              @Override
              public void onAdFailedToLoad(@NonNull LoadAdError error) {
                ...
              }
            }).build();
```

5.2 load native ad

```
adLoader.loadAd(new AdRequest.Builder().build());
```

ProGuard setting

you should add the following rules in proguard-project.txt , or you may not run sdk pass through .

```
-keep public class android.webkit.JavascriptInterface { *; }
-dontwarn com.iab.omid.library.adview.**
-keep public class com.iab.omid.library.adview.**.* { *; }

-dontwarn com.advg.**
-keep public class com.advg.**.* { *; }
```

and also, adview's adapter codes also not be obfuscated, adview 's custom adapter class's package path is :

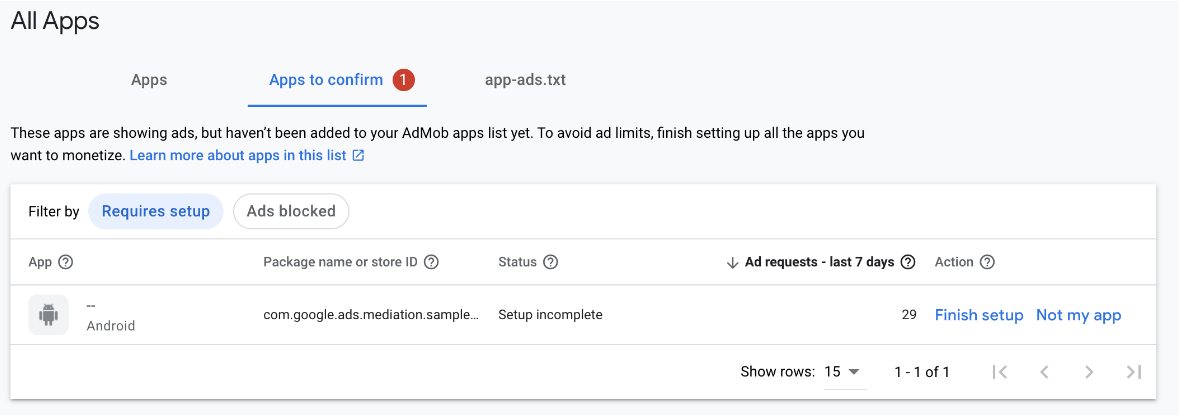
```
package com.google.ads.mediation.adview.customevent.AdVGCustomEvent;
```

so you must declare the following definition in proguard-project.txt :

```
-keep public class com.google.ads.mediation.adview.customevent { *; }
```

Enable debug tracking

Starting September 2022, google will start limiting ad serving to apps that haven't been reviewed and approved, so begin the review process, pls Check your 'Apps to confirm' page. check 'Apps to confirm' for apps that require additional setup. If there aren't any apps on your 'Apps to confirm' page, no action is needed.

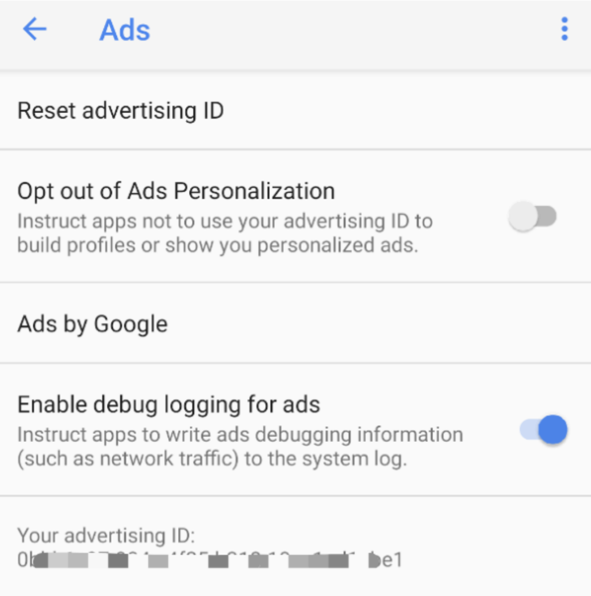


These apps are showing ads, but haven't been added to your AdMob apps list yet. To avoid ad limits, finish setting up all the apps you want to monetize. Learn more about apps in this list

App	Package name or store ID	Status	Ad requests - last 7 days	Action
-- Android	com.google.ads.mediation.sample...	Setup incomplete	29	Finish setup Not my app

Show rows: 15 1 - 1 of 1

Also, if you don't want confirm app yet, you can test you app adapter with debug mode. To enable network tracing, you will need to enable developer options for your device. Launch the Google Settings app and select **Google > Ads > Enable debug** logging for ads. A more detailed guide is available for both AdMob and Google Ad Manager publishers.



Reset advertising ID

Opt out of Ads Personalization ○
Instruct apps not to use your advertising ID to build profiles or show you personalized ads.

Ads by Google

Enable debug logging for ads ●
Instruct apps to write ads debugging information (such as network traffic) to the system log.

Your advertising ID:
0[redacted]e1

You're all done!

Your AdMob SDK should start showing AdView ads immediately.

Otherwise please contact your AM or partner@adview.com.