AdMob Adapter Installation

Android Admob Adapter Installation

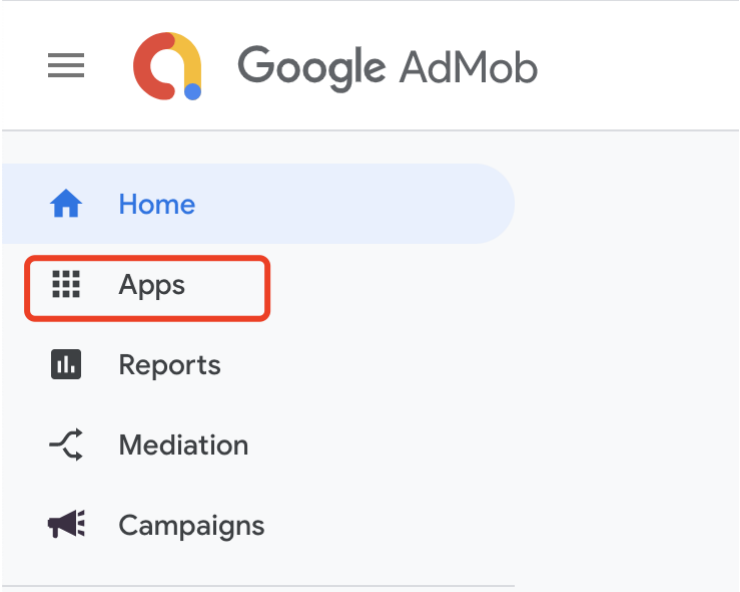
Adding AdView to AdMob

Step 1: Login to your AdMob account and configure AdView

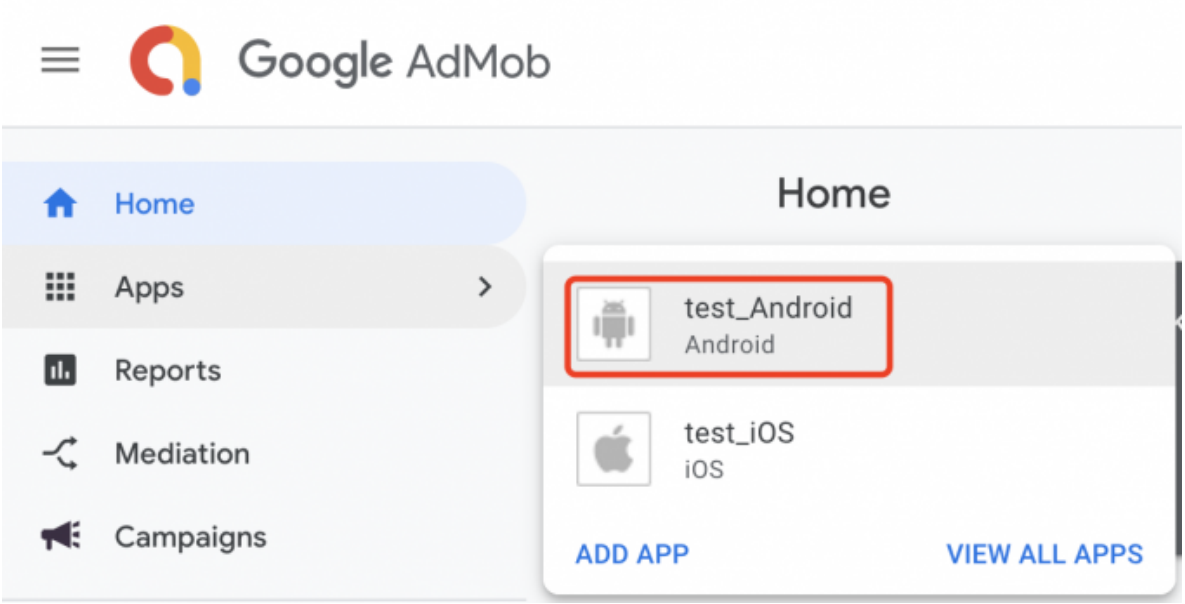
As explained above, you must have a AdMob account, and an associated app within this account, in order to integrate AdView.

Step 2: Select your app and click to monetize it

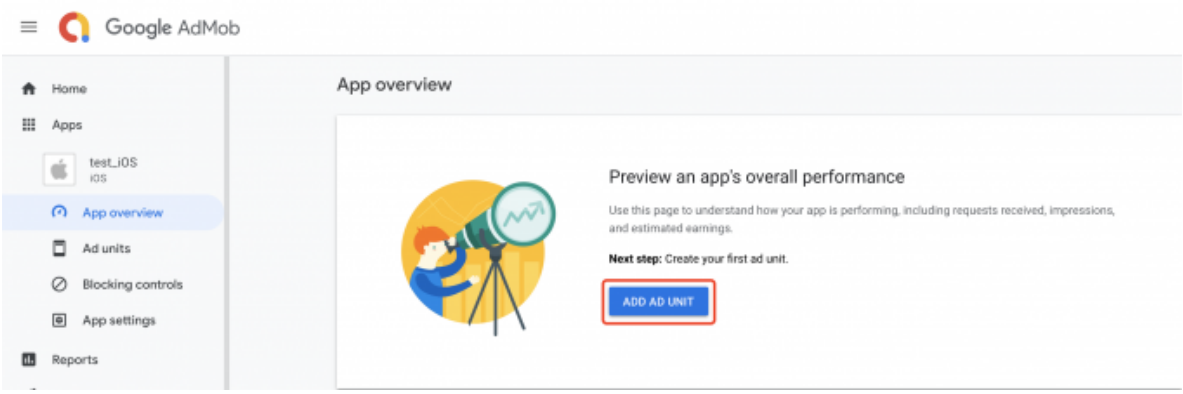
Within your AdMob account, press the 'Apps' tab



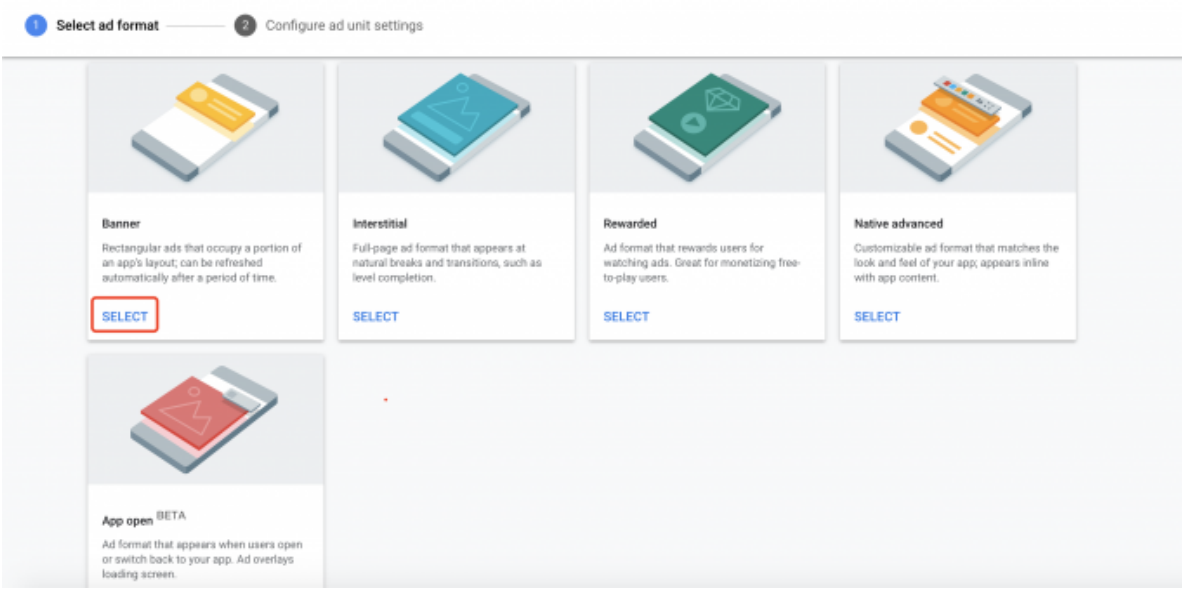
AdView must be integrated as a custom event. If your app does not have a ad unit yet, you should create a new ad unit by clicking on your app's name:



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



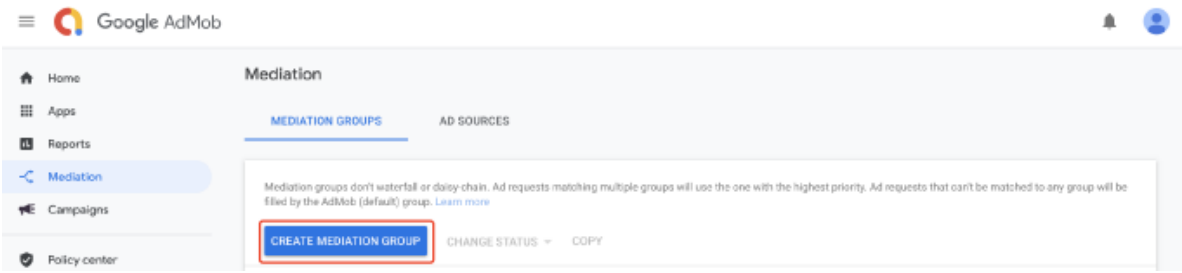
Provide a name for your new ad unit. After making the device type selection, you should choose a format that you need to installation:



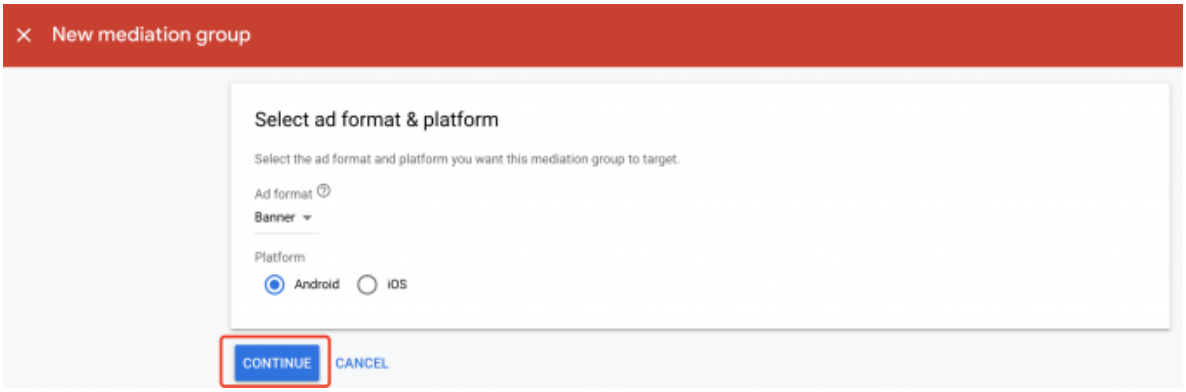
note: create MREC AD unit, you need select "Banner". The configuration is the same as Banner.

Step 3: Create Mediation group for your ad placement

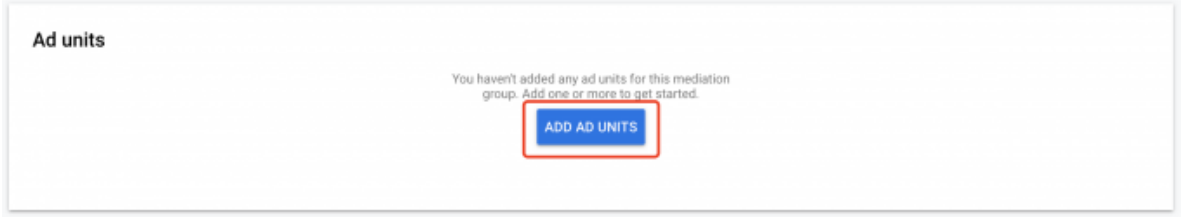
Within your AdMob account, press the 'Mediation' tab,and click "Create Mediation group".



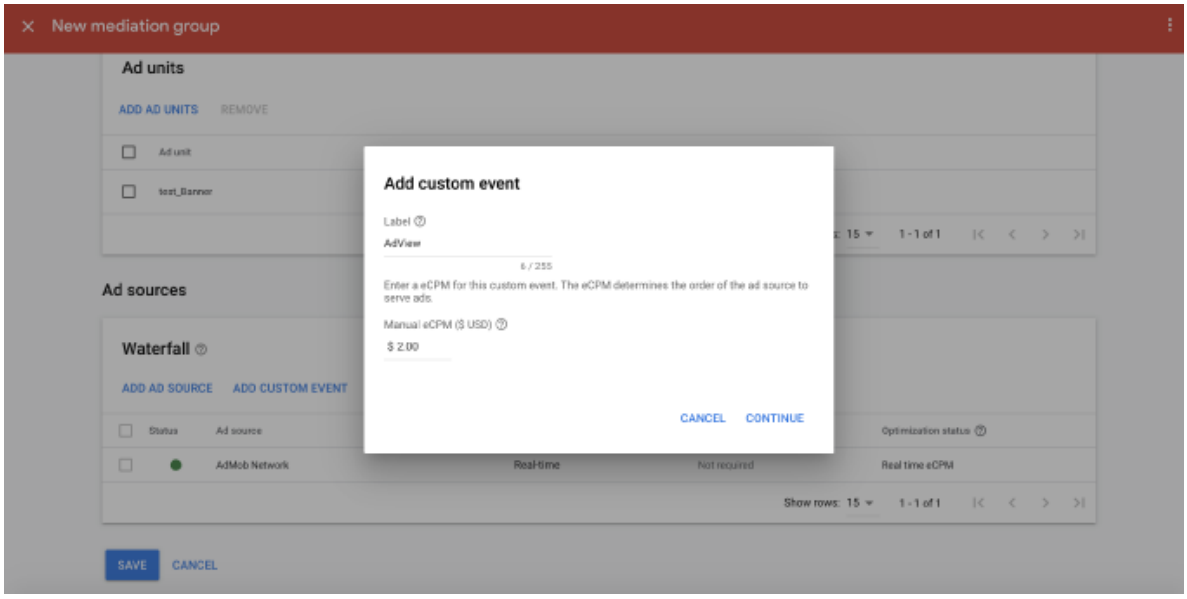
Then click ''CONTINUE''



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

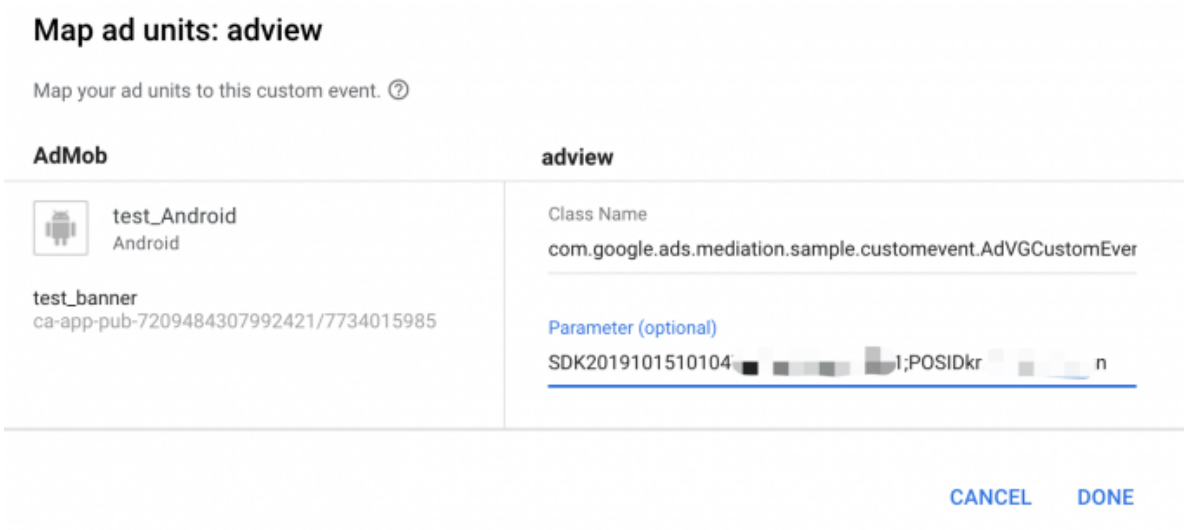


AdView must be integrated as a custom event,so please click Add Custom Event.When you finish write Label and eCPM click continue



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

(pic 3-1)



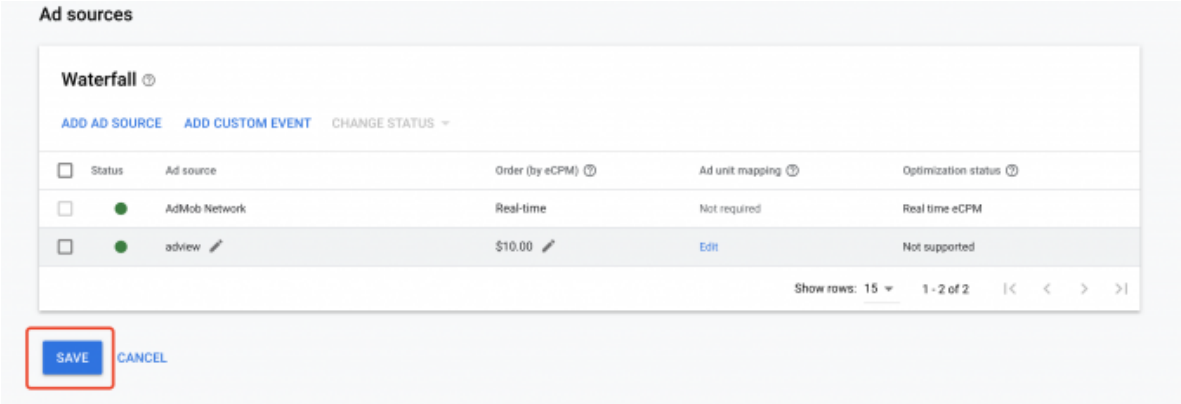
Class Name

```
com.google.ads.mediation.sample.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'



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:

- AdView SDK
- AdViewSDK_AdMobAdapter_demo

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

Step 2: Add the AdView SDK into your Project

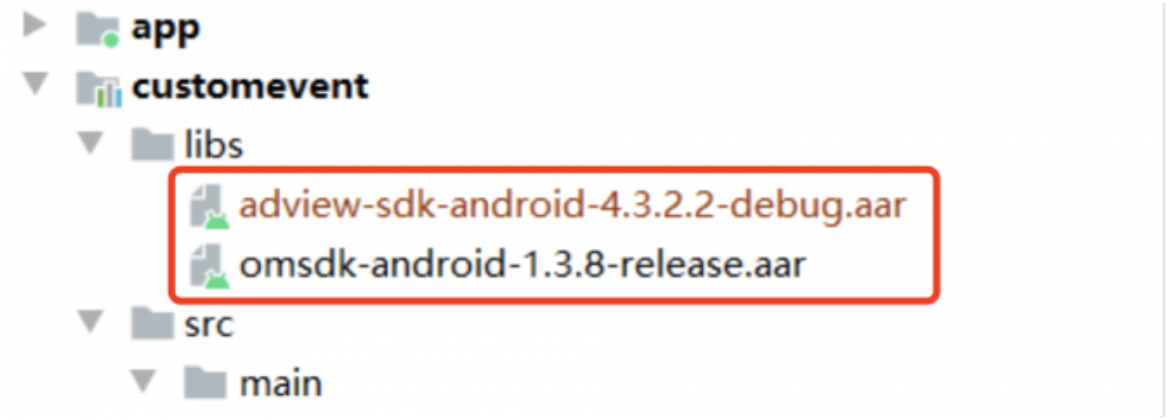
first, add module 'customevent' to build.gradle

```
dependencies {
    implementation project(':customevent')
}
```

Ad the following code to the project's settings.gradle

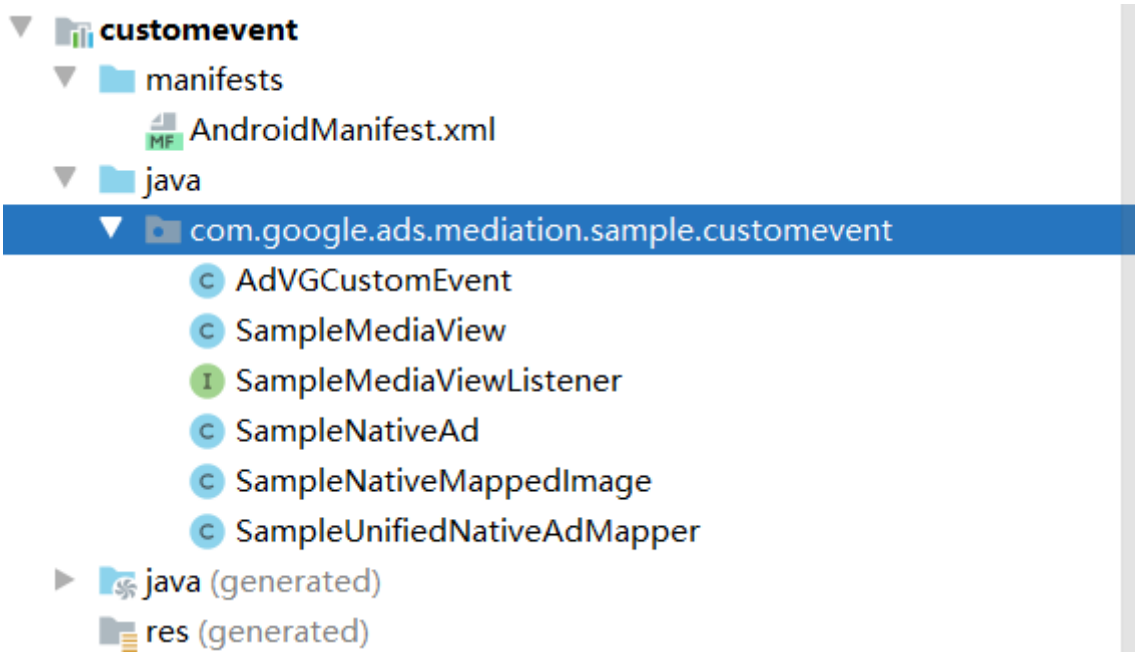
```
include ':customevent'
```

next, add adview's sdk libs into customevent module:



And then , we provide a sample reference codes for CustomEvent for adview's sdk invoking.

you can find the following files in correspond directory:/AdViewSDK_AdMobAdapter_demo /Example/customevent



note: the package name must be same as you set in admob ads server setting (see pic3-1)

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

```
    // Sample custom event banner.
    AdView mCustomEventAdView = (AdView) findViewById(R.id.customevent_adview);
//your unitkey id
    mCustomEventAdView.loadAd(new AdRequest.Builder().build());
```

2. interstitial

```
     /**
     * Sample Custom Event.
     * 1) Create the sample custom event interstitial.
     * 2) Set up the on click listener for the sample custom event interstitial
button.
```

```
     * 3) Create the sample custom event interstitial.
     */
    customEventInterstitial = new InterstitialAd(this);

 customEventInterstitial.setAdUnitId(getResources().getString(R.string.interstit
ial_ad_unit_id_advg));
    //set listener
    customEventInterstitial.setAdListener(new AdListener() {
      @Override
      public void onAdFailedToLoad(int errorCode) {
        customEventInterstitialButton.setEnabled(true);
      }
      ... //some codes (see demo)
    });
    customEventInterstitial.loadAd(new AdRequest.Builder().build());
```

3. Reward

3.1 load reward

```
    customEventRewardedAd = new RewardedAd(this,
getString(R.string.rewarded_ad_unit_id_advg)); //unitkey
    customEventRewardedAd.loadAd(new AdRequest.Builder().build(),
adLoadCallback); //callback add
```

3.2 show reward

```
    customEventRewardedAd.show(MainActivity.this, adCallback);
```

3.3 reward callback

```
    RewardedAdCallback adCallback = new RewardedAdCallback() {
          @Override
          public void onRewardedAdOpened() {
            Toast.makeText(MainActivity.this,
                "Rewarded ad opened",
                Toast.LENGTH_SHORT).show();
          }
          @Override
          public void onRewardedAdClosed() {
            requestCustomEventRewardedAd(); //pre load video
          }
        ... //more see demo
        };
```

4. Native

```
     // Sample Custom Event Native ad.
    customEventNativeLoader = new AdLoader.Builder(this,
        getResources().getString(R.string.customevent_native_ad_unit_id_advg))
//unitId
        .forUnifiedNativeAd(new
UnifiedNativeAd.OnUnifiedNativeAdLoadedListener() {
          @Override
          public void onUnifiedNativeAdLoaded(UnifiedNativeAd unifiedNativeAd) {
```

```
            FrameLayout frameLayout =
                (FrameLayout) findViewById(R.id.customeventnative_framelayout);
            UnifiedNativeAdView adView = (UnifiedNativeAdView)
getLayoutInflater()
                .inflate(R.layout.ad_unified, null);
            populateUnifiedNativeAdView(unifiedNativeAd, adView);
            frameLayout.removeAllViews();
            frameLayout.addView(adView);
          }
        })
        .withAdListener(new AdListener() {
          @Override
          public void onAdFailedToLoad(int errorCode) {
            Toast.makeText(MainActivity.this,
                "Custom event native ad failed with code: " + errorCode,
                Toast.LENGTH_SHORT).show();
          }
        }).build();
    //after click load button, it will trigger native ad loading & show
    Button refreshCustomEvent = (Button)
findViewById(R.id.customeventnative_button);
    refreshCustomEvent.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View unusedView) {
        customEventNativeLoader.loadAd(new AdRequest.Builder().build()); //load
native ad
      }
    });
```

Step 4: CustomEvent adapter module

And you can see the custom event codes for AdView in AdVGCustomEvent.java. this is a main entry for AdMob's custom event . and you will find 4 types ads entry class :

```
public class AdVGCustomEvent extends Adapter implements CustomEventBanner,
        CustomEventInterstitial,
        CustomEventNative,
        MediationRewardedAd)
```

also the following different ad types of override functions:

```
@Override
    public void requestBannerAd(final Context context,
                                CustomEventBannerListener listener,
                                String serverParameter,
                                AdSize size,
                                MediationAdRequest mediationAdRequest,
                                Bundle customEventExtras)
```

```
@Override
    public void requestInterstitialAd(Context context,
                                      CustomEventInterstitialListener listener,
                                      String serverParameter,
                                      MediationAdRequest mediationAdRequest,
                                      Bundle customEventExtras)
```

```
@Override
    public void requestNativeAd(Context context,
                                CustomEventNativeListener
customEventNativeListener,
                                String serverParameter,
                                NativeMediationAdRequest
nativeMediationAdRequest,
                                Bundle extras)
```

```
@Override
    public void loadRewardedAd( MediationRewardedAdConfiguration
mediationRewardedAdConfiguration,
                                MediationAdLoadCallback<MediationRewardedAd,
MediationRewardedAdCallback>
 mediationAdLoadCallback)
```

in this class, adview's implementation has been done already, but if you want to changed it for your customize needed. you can do your changes with AdView's sdk menu to handle it .

Step 5: GDPR

As a publisher, you should integrate a Consent Management Platform (CMP) and request for vendor and purpose consents as outlined in IAB Europe's Mobile In-App CMP API v1.0: Transparency & Consent Framework. You can find a reference implementation of a web-based CMP and the corresponding native wrappers here in the IAB's GDPR Transparency and Consent Framework.
if you already got IABConsent_ConsentString, you just use setGDPR2() method to provide gdpr subjects in AdVGCustomEvent.java, and you will see the following codes in multi places:

prototype:

```
public void setGDPR2(int gdprApplies, String consentString);
```

for example:

```
adViewBIDView.setGDPR2(1,Gdpr); //Gdpr means consent string
```

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.kuaiyou.**
-keep public class com.kuaiyou.**.* { *; }
-keep public class com.AdVG.**.* { *; }
```

and also, all SDK listeners should not be obfuscated, for a listener callback example :

```
package com.abc.adviewdriver;
class MyListener implements
com.kuaiyou.loader.loaderInterface.AdViewInstlListener {
        public void onAdClicked() {
                ...
        }
        ...
}
```

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

```
-keep com.abc.adviewdriver.MyListener { public void on*(); }
```

You’re all done!

Your AdMob SDK should start showing AdView ads immediately.

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