**SuperADS advertising SDK integration - Android**

SuperADS advertising SDK provides five different Ads formats: Banner, Interstitial, Native, Reward Video, Playable Ads. This document instruction you how to integrate the SDK for Android developers.

To take a look at Ad showing, get “asdian” APP from Google Play:

<https://play.google.com/store/apps/details?id=com.asdian.app>

code sample:

<https://github.com/156076769/superads_standalone_demo>

1. **Preparation, get a PublisherId for u and get a ApplicationId for your App**

SuperADS assign a unique id to each developer, which we call it PublisherId. Also for a APP of this developer, we assign a unique id to it, which we call it ApplicationId. Currently our operation staff will give you them, later you can login [www.superads.cn/xxx](http://www.superads.cn/xxx) to create them by yourself later.

There is also a publick key(xxx.pem) needed when your APP communicates with SuperADS server. Please contact our operator to get this key file.

1. **Dependence to SDK and Initial it**

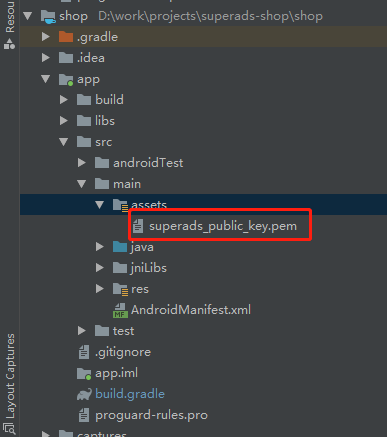
2.1 We already publish SDK library to jcenter, in your APP build.gradle, add follow to depend it

1. implementation 'com.superads.android:adsdk:0.1.2'
2. implementation 'com.github.bumptech.glide:glide:4.9.0'
3. implementation 'com.squareup.retrofit2:retrofit:2.5.0'
4. implementation 'com.squareup.retrofit2:converter-gson:2.5.0'

2.2 Proguard. Follow is the class, members, method that can’t be confused, please add them in your APP proguard rule

1. -keep **class** com.bumptech.\*\* { \*; }
2. -keep **public** **class** \* **implements** com.bumptech.glide.module.GlideModule
3. -keep **public** **enum** com.bumptech.glide.load.resource.bitmap.ImageHeaderParser$\*\* {
4. \*\*[] $VALUES;
5. **public** \*;
6. }
7. -keep **class** okhttp3.\*\* { \*; }
8. -keep **class** retrofit2.\*\* { \*; }
9. -keep **class** com.superads.android.adsdk.ads.utils.CryptoUtils { \*; }
10. -keep **class** com.superads.android.adsdk.ads.providers.SuperAds { \*; }
11. -keep **class** com.superads.android.adsdk.ads.providers.models.AdRequest { \*; }
12. -keep **class** com.superads.android.adsdk.ads.providers.models.AdRequest$Builder { \*; }
13. -keep **class** com.superads.android.adsdk.ads.rendering.view.BannerAdView { \*; }
14. -keep **class** com.superads.android.adsdk.ads.rendering.view.InterstitialAdLoader { \*; }
15. -keep **class** com.superads.android.adsdk.ads.providers.models.NativeAdRequest { \*; }
16. -keep **class** com.superads.android.adsdk.ads.providers.models.NativeAdRequest$Builder { \*; }
17. -keep class com.superads.android.adsdk.ads.providers.models.NativeAdRequest$OnNativeAdLoadedListener { \*; }
18. -keep class com.superads.android.adsdk.ads.providers.models.NativeAd { \*; }
19. -keep **class** com.superads.android.adsdk.ads.rendering.view.NativeAdLoader { \*; }
20. -keep **class** com.superads.android.adsdk.ads.rendering.view.VideoAdLoader { \*; }
21. -keep **class** com.superads.android.adsdk.ads.rendering.view.AdListener { \*; }
22. -keep **class** com.superads.android.adsdk.ads.providers.SuperAdsConfig { \*; }

2.3 As talk above, a pem key file is need when APP communicates with SuperADS server. This key is a RSA public key to encrypt some sensitive info in request and response. So when your get it , put it in APP assets folder, and named it “superads\_publick\_key.pem”



2.4 SDK initializtion

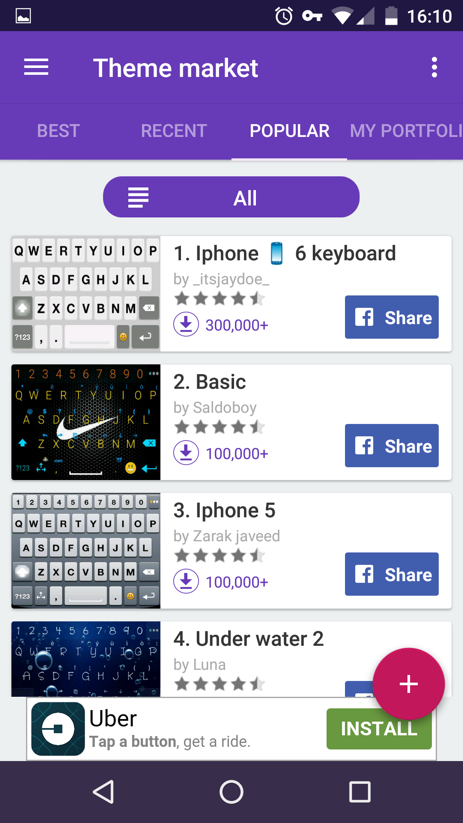
1. SuperAds.init(context,
2. "your publisherId", "your applicationId");

We suggest you invoke it in APP application onCreate()

1. **Request an ad and show it**

**3.1 Banner Ads**

Banner Ad is 320\*50 size picture that show in Screen top or bottom, while if screen orientation landscape, it’s a 768\*90 picture. A example is below.



How to add this ad to your app?

Step1: add a FrameLayout to contain this picture in your activity layout, and place it in top or bottom

1. <com.superads.android.adsdk.ads.rendering.view.BannerAdView
2. android:id="@+id/card\_banner"
3. android:layout\_width="wrap\_content"
4. android:layout\_height="wrap\_content"
5. android:layout\_gravity="center" />

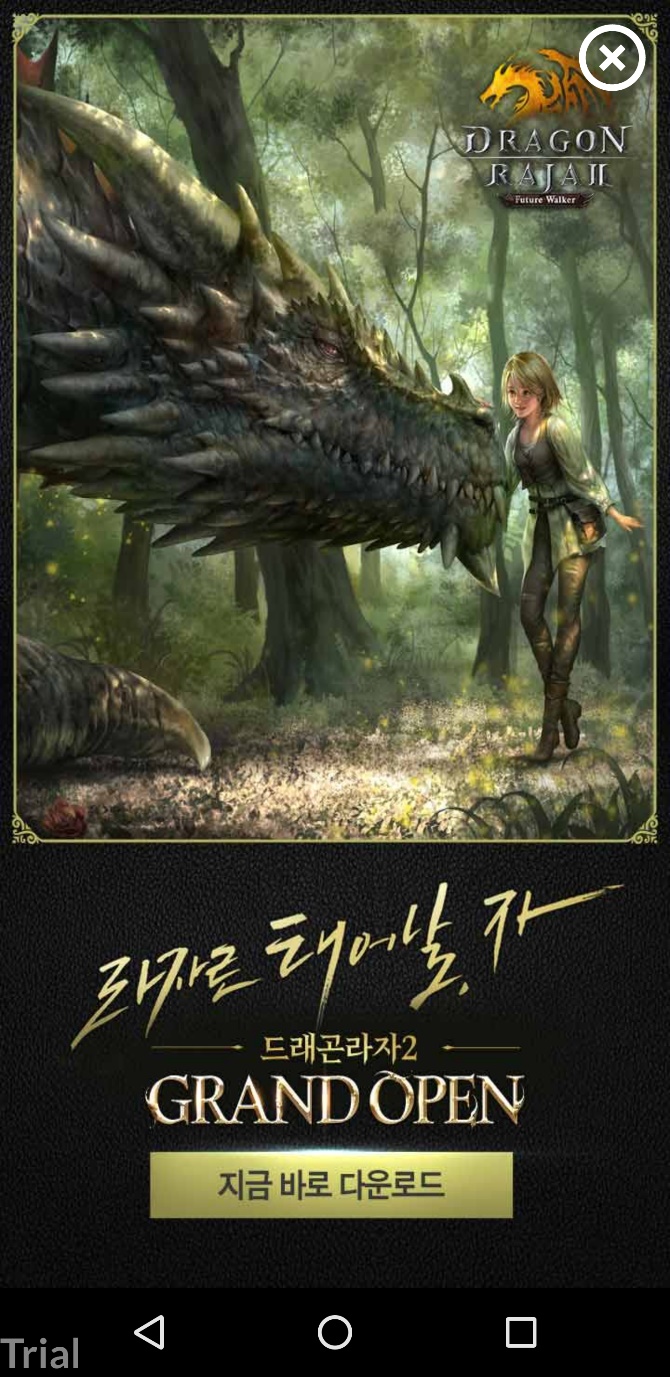
Step2: call java API to request SuperADS server, when response received, a callback will invoked and you could show view at this time.

1. Button btnBanner = findViewById(R.id.btn\_load);
2. btnBanner.setOnClickListener(view -**>** {
3. final BannerAdView bannerAdView = findViewById(R.id.card\_banner);
4. AdRequest.Builder builder = new AdRequest.Builder(SuperAds.genRandomPlacementId());
5. bannerAdView.loadAd(builder.build(), new AdListener() {
6. @Override
7. public void onAdLoaded() {
8. bannerAdView.setVisibility(View.VISIBLE);
9. }
11. @Override
12. public void onAdFailedToLoad(int errorCode) {
13. Log.e("NativeExamplesActivity", "error generating ad, error code=" + errorCode);
14. }
15. });
16. });

A placement id is need when build a AdRequest, if you don’ have , you can call SuperAds.genRandomPlacementId() to generate a ramdom one.

**3.2 Interstitial Ads**

Interstitial Ad is a big picture show in full screen when opening APP or jump to another page. Below is an example.



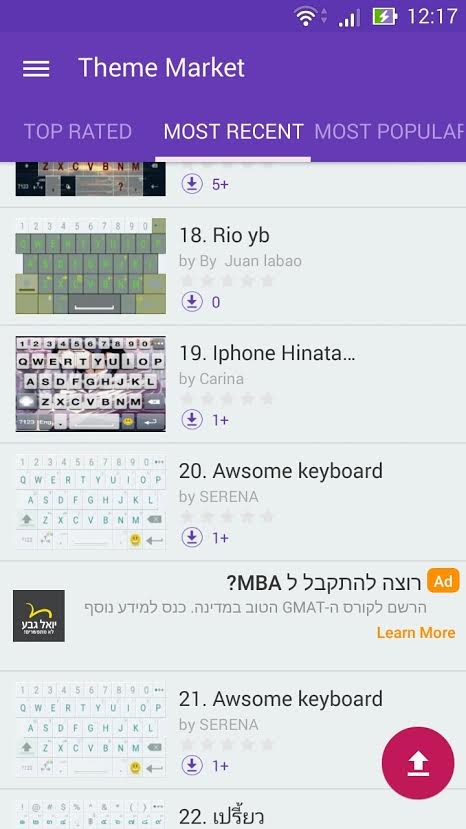
How to request and show it?

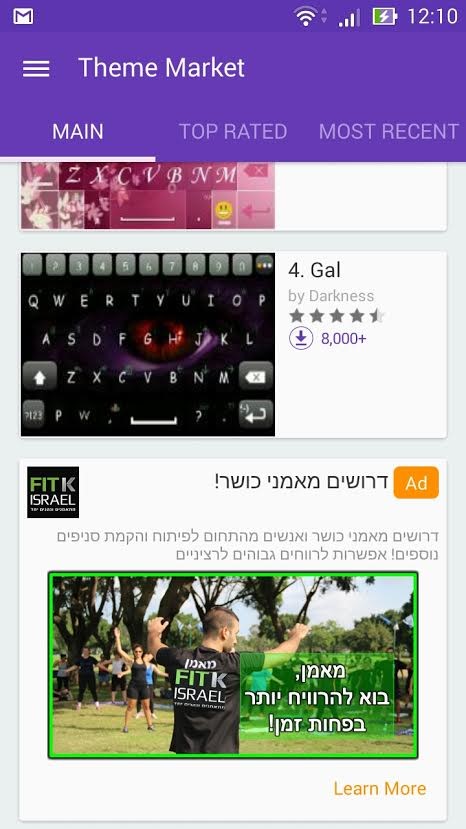
Before you jump to an activity, invoke like bellow, a full screen picture will show. After user close Ad, onAdClosed callback will invoke, here you can jump to your activity

1. Button btnLoad = findViewById(R.id.btn\_load);
2. btnLoad.setText("Load Interstitial");
3. btnLoad.setOnClickListener(view -**>** {
4. AdRequest.Builder builder = new AdRequest.Builder(SuperAds.genRandomPlacementId());
5. this.interstitialAd = new InterstitialAdLoader(this);
6. interstitialAd.loadAd(builder.build(), new AdListener() {
7. @Override
8. public void onAdLoaded() {
9. Toast.makeText(InterstitialActivity.this, "interstitial Ad load successfully", Toast.LENGTH\_SHORT).show();
10. }
12. @Override
13. public void onAdFailedToLoad(int errorCode) {
14. Log.e("NativeExamplesActivity", "error generating ad, error code=" + errorCode);
15. }
17. @Override
18. public void onAdClosed() {
19. super.onAdClosed();
20. }
21. });
22. });
24. View btnShow = findViewById(R.id.btn\_show);
25. btnShow.setOnClickListener(view -**>** {
26. if (interstitialAd != null && interstitialAd.isloaded())
27. interstitialAd.show();
28. });

**3.3 Native Ads**

Native Ad is a RecycleView item that embed to origin list. It shows like the origin content and disturb less to user. This format is a popular currently, we provides two , Native Banner and Native Feed. Below it’s an example.





When request a Native Ad, SuperADS server responses 4(for Native Feed) or 3(for Native Banner) elements to you, they are

1. APP Icon：a small image that represent app icon



2. APP Title：app name



3. APP Description：description of this app, less than 120 words



4. APP Big Image：big image for APP，only Native Feed have it



Developer compose these elements together, place them appropriately, build a recycleView item, make the style same as the origin item, and then insert it to origin list.

Bellow is a example.

**Step1: define a item layout that same as the origin recycleView item, contains the 4(3) elements that SuperADS server response.**

1. **<?xml** version="1.0" encoding="utf-8"**?>**
2. **<LinearLayout** xmlns:android="http://schemas.android.com/apk/res/android"
3. android:layout\_width="match\_parent"
4. android:layout\_height="wrap\_content"**>**
6. **<FrameLayout**
7. android:id="@+id/adContainer"
8. android:layout\_width="match\_parent"
9. android:layout\_height="wrap\_content"**/>**
11. **</LinearLayout>**
12. **<?xml** version="1.0" encoding="utf-8"**?>**
13. **<RelativeLayout** xmlns:android="http://schemas.android.com/apk/res/android"
14. xmlns:tools="http://schemas.android.com/tools"
15. android:layout\_width="match\_parent"
16. android:layout\_height="wrap\_content"
17. android:layout\_gravity="center"
18. android:layout\_margin="0dp"
19. android:background="@color/white"
20. android:padding="0dp"**>**
22. **<TextView**
23. android:id="@+id/ad\_txt\_ad\_indicator"
24. android:layout\_width="wrap\_content"
25. android:layout\_height="wrap\_content"
26. android:layout\_alignParentTop="true"
27. android:layout\_alignParentEnd="true"
28. android:layout\_alignParentRight="true"
29. android:layout\_margin="1dp"
30. android:gravity="center"
31. android:text="Sponsored"
32. android:textColor="@color/blue"
33. android:textSize="12sp"
34. tools:ignore="HardcodedText" **/>**
36. **<ImageView**
37. android:id="@+id/ad\_img\_icon"
38. android:layout\_width="wrap\_content"
39. android:layout\_height="wrap\_content"
40. android:layout\_alignTop="@+id/ad\_txt\_title"
41. android:layout\_alignBottom="@+id/ad\_txt\_title"
42. android:layout\_margin="5dp"
43. android:scaleType="fitXY" **/>**
45. **<TextView**
46. android:id="@+id/ad\_txt\_title"
47. android:layout\_width="wrap\_content"
48. android:layout\_height="wrap\_content"
49. android:layout\_below="@+id/ad\_txt\_ad\_indicator"
50. android:layout\_toEndOf="@+id/ad\_img\_icon"
51. android:layout\_toRightOf="@+id/ad\_img\_icon"
52. android:padding="4dp"
53. android:textColor="#006064"
54. android:textSize="24sp"
55. tools:text="Awesome application that you have to try is now available for free download" **/>**
57. **<ImageView**
58. android:id="@+id/ad\_img"
59. android:layout\_width="match\_parent"
60. android:layout\_height="150dp"
61. android:layout\_below="@+id/ad\_txt\_title"
62. android:layout\_centerHorizontal="true"
63. android:layout\_margin="8dp"
64. android:scaleType="fitXY" **/>**
66. **<TextView**
67. android:id="@+id/ad\_txt\_description"
68. android:layout\_width="wrap\_content"
69. android:layout\_height="wrap\_content"
70. android:layout\_below="@+id/ad\_img"
71. android:layout\_margin="12dp"
72. android:ellipsize="end"
73. android:padding="2dp"
74. android:textColor="#009688"
75. android:textSize="16sp"
76. tools:text="Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat." **/>**
78. **<TextView**
79. android:id="@+id/ad\_txt\_cta"
80. android:layout\_width="wrap\_content"
81. android:layout\_height="wrap\_content"
82. android:layout\_below="@id/ad\_txt\_description"
83. android:layout\_alignParentEnd="true"
84. android:layout\_alignParentRight="true"
85. android:layout\_margin="8dp"
86. android:background="@drawable/cta\_square\_green"
87. android:gravity="center"
88. android:paddingStart="10dp"
89. android:paddingLeft="10dp"
90. android:paddingTop="5dp"
91. android:paddingEnd="10dp"
92. android:paddingRight="10dp"
93. android:paddingBottom="5dp"
94. android:textAllCaps="true"
95. android:textColor="@color/white"
96. android:textSize="24sp"
97. android:textStyle="bold"
98. tools:text="Download now" **/>**
100. **</RelativeLayout>**

**Step2: build a ViewHolder for Native Ad that will insert into recycleView adapter. When building, asynchronous request Ad, add ViewHodler view to recycleView when a response callback invoked.**

1. private void loadNativeAds() {
2. NativeAdRequest.Builder builder = new NativeAdRequest.Builder(SuperAds.genRandomPlacementId(), new NativeAdRequest.OnNativeAdLoadedListener() {
3. @Override
4. public void onNativeAdLoaded(NativeAd nativeAd) {
5. refreshRecycleView(nativeAd);
6. }
7. });
9. final NativeAdLoader nativeAd = new NativeAdLoader(NativeActivity.this);
10. nativeAd.loadAd(builder.build(), new AdListener() {
11. @Override
12. public void onAdLoaded() {
13. Log.i("NativeAdapter", "native ad loaded");
14. }
16. @Override
17. public void onAdFailedToLoad(int errorCode) {
18. Log.e("NativeAdapter", "error generating ad, error code=" + errorCode);
19. }
20. });
21. }
23. private void refreshRecycleView(NativeAd nativeAd) {
24. DataType dataType = DataType.NATIVE\_BANNER;
25. if (getIntent().hasExtra("adType")) {
26. dataType = (DataType) getIntent().getSerializableExtra("adType");
27. }
29. List**<ItemData>** items = new ArrayList**<>**();
30. for (int i = 0; i **<** **10**; i++) {
31. ItemData data = new ItemData();
32. data.type = 1;
33. data.data = "txt";
34. items.add(data);
35. }
36. ItemData adItem = new ItemData();
37. adItem.type = 100;
38. adItem.data = nativeAd;
39. items.add(3, adItem);
41. switch (dataType) {
42. case NATIVE\_BANNER:
43. reycler.setAdapter(new NativeAdapter(items));
44. break;
45. case NATIVE\_FEED:
46. default:
47. reycler.setAdapter(new NativeAdapter(items));
48. break;
49. }
50. }
51. @Override
52. public RecyclerView.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
53. View v;
54. if (viewType == NORMAL) {
55. v = LayoutInflater.from(parent.getContext())
56. .inflate(R.layout.list\_item, parent, false);
57. return new ItemViewHolder(v);
58. } else {
59. final View adContainer = LayoutInflater.from(parent.getContext()).inflate(R.layout.superads\_native\_ad\_container, null);
60. final View adView = LayoutInflater.from(parent.getContext()).inflate(R.layout.superads\_native\_ad, null);
61. return new NativeAdViewHolder(adContainer, adView);
62. }
63. }
65. @Override
66. public void onBindViewHolder(@NonNull RecyclerView.ViewHolder holder, int position) {
67. ItemData data = list.get(position);
68. if (data.type != 100) {
69. String txt = (String) data.data;
70. ((ItemViewHolder) holder).bindData(txt);
71. } else {
72. NativeAd nativeAd = (NativeAd)data.data;
73. NativeAdViewHolder nativeAdViewHolder = (NativeAdViewHolder)holder;
74. nativeAdViewHolder.bindData(nativeAd);
75. }
76. }
77. public class ItemViewHolder extends RecyclerView.ViewHolder {
79. private TextView txtView;
81. public ItemViewHolder(@NonNull View v) {
82. super(v);
83. txtView = v.findViewById(R.id.tvTitle);
84. }
86. public void bindData(String txt) {
87. txtView.setText(txt);
88. }
89. }
91. public class NativeAdViewHolder extends RecyclerView.ViewHolder {
92. private ViewGroup adContainer;
93. private ViewGroup adView;
95. private TextView titleTextView;
96. private TextView descriptionsTextView;
97. private TextView callToActionTextView;
98. private ImageView iconImageView;
99. private ImageView bigImageImageView;
101. public NativeAdViewHolder(@NonNull View itemView, View adView) {
102. super(itemView);
104. this.adContainer = itemView.findViewById(R.id.adContainer);
105. this.adView = (ViewGroup)adView;
106. this.adContainer.addView(this.adView);
108. this.titleTextView = itemView.findViewById(R.id.ad\_txt\_title);
109. this.descriptionsTextView = itemView.findViewById(R.id.ad\_txt\_description);
110. this.callToActionTextView = itemView.findViewById(R.id.ad\_txt\_cta);
111. this.iconImageView = itemView.findViewById(R.id.ad\_img\_icon);
112. this.bigImageImageView = itemView.findViewById(R.id.ad\_img);
113. }
115. public void bindData(NativeAd nativeAd) {
116. nativeAd.renderTitle(titleTextView);
117. nativeAd.renderDesc(descriptionsTextView);
118. nativeAd.renderCta(callToActionTextView);
119. nativeAd.renderIcon(iconImageView);
120. nativeAd.renderBigImgUrl(bigImageImageView);
121. nativeAd.registerViewForInteraction(adView);
122. }
123. }

**3.4 Video Ads**

Video Ads is a short video that plays to user when jumping Activities, it will show in full screen.

1. Button btnLoad = findViewById(R.id.btn\_load);
2. btnLoad.setText("Load Video");
3. btnLoad.setOnClickListener(view -**>** {
4. AdRequest.Builder builder = new AdRequest.Builder(SuperAds.genRandomPlacementId());
5. this.videoAdLoader = new VideoAdLoader(this);
6. videoAdLoader.loadAd(builder.build(), new AdListener() {
7. @Override
8. public void onAdLoaded() {
9. Toast.makeText(VideoActivity.this, "video Ad load successfully", Toast.LENGTH\_SHORT).show();
10. }
12. @Override
13. public void onAdFailedToLoad(int errorCode) {
14. Log.e("NativeExamplesActivity", "error generating ad, error code=" + errorCode);
15. }
17. @Override
18. public void onAdClosed() {
19. super.onAdClosed();
20. }
21. });
22. });
24. View btnShow = findViewById(R.id.btn\_show);
25. btnShow.setOnClickListener(view -**>** {
26. if (videoAdLoader != null && videoAdLoader.isloaded())
27. videoAdLoader.show();
28. });