

Mobile mediation

Integration

10.07.2024

Mobile mediation. Integration. Version 2.0

Document build date: 10.07.2024

This volume is a part of Yandex technical documentation.

© 2008—2024 Yandex LLC. All rights reserved.

Copyright Disclaimer

Yandex (and its applicable licensor) has exclusive rights for all results of intellectual activity and equated to them means of individualization, used for development, support, and usage of the service Mobile mediation. It may include, but not limited to, computer programs (software), databases, images, texts, other works and inventions, utility models, trademarks, service marks, and commercial denominations. The copyright is protected under provision of Part 4 of the Russian Civil Code and international laws.

You may use Mobile mediation or its components only within credentials granted by the Terms of Use of Mobile mediation or within an appropriate Agreement.

Any infringements of exclusive rights of the copyright owner are punishable under civil, administrative or criminal Russian laws.

Contact information

Yandex LLC

<https://www.yandex.com>

Ten.: +7 495 739 7000

Email: pr@yandex-team.ru

16 L'va Tolstogo St., Moscow, Russia 119021

Contents

Enabling the plugin.....	4
Ad formats.....	6
Banner ads.....	6
Enabling a banner.....	6
Example of working with banner ads.....	6
Interstitial ads.....	7
Creating and displaying InterstitialAd.....	7
Example of working with interstitial ads.....	8
Rewarded ads.....	8
Creating and displaying RewardedAd.....	8
Example of working with rewarded ads.....	9
Mobile mediation adapters.....	9
AdMob.....	10
myTarget.....	10
Start.io.....	11
UnityAds.....	11
AppLovin.....	11
IronSource.....	12
AdColony.....	12
ChartBoost.....	12
Pangle.....	12
Tapjoy.....	13
Vungle.....	13
Mintegral.....	13

Adding the Yandex Mobile Ads Flutter plugin



Warning:

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

About the plugin

Flutter is a popular technology for creating mobile apps. Developers use it to write single code that can be run on different operating systems, including Android and iOS.

Now, it will be easier for you to monetize Android and iOS apps created with Flutter. All our partners in the Yandex Advertising Network can now use the plugin to enable Yandex-based monetization in their apps.

This plugin supports loading and display of the following ad types:

- [Banner ads](#)
- [Interstitial ads](#)
- [Rewarded ads](#).

Requirements

- Flutter 2.5.0 or higher.
- Android 4.1 or higher.
 - Video ads are selected for devices with Android 5.0 or higher.
- iOS 12.0 or higher.
 - Check the [additional steps](#) to ensure that iOS 14 (or higher) works properly.
 - Latest Xcode version with enabled command-line tools.
- [Registering](#) an account in the Yandex Advertising Network.

Adding the plugin to your project

Install the Yandex Mobile Ads Flutter plugin in your project. From the root of the project, call the command:

```
flutter pub add yandex_mobileads
```

Once the plugin is added, you'll see the following line with a dependency in the pubspec .yaml file:

```
dependencies:  
  yandex_mobileads: ^X.X.X
```

X.X.X: The plugin's [current version](#) number.

Configuring for specific platforms

Android

1. In the app/build.gradle file of your app's Android module, add:

```
dependencies {  
  // ...  
  implementation 'com.yandex.android:mobileads-mediation:5.10.0.0'  
}
```

2. In the build.gradle file of your app's Android module, add:

```
// IronSource  
maven {  
  url 'https://android-sdk.is.com/'  
}  
  
// Pangle  
maven {  
  url 'https://artifact.bytedance.com/repository/pangle'  
}
```

```
// Tapjoy
maven {
    url "https://sdk.tapjoy.com/"
}

// Mintegral
maven {
    url "https://dl-maven-android.mintegral.com/repository/mbridge_android_sdk_oversea"
}
```

- The new permission `com.google.android.gms.permission.AD_ID` is already added in the Yandex Mobile Ads SDK. It enables you to use `AD_ID` to select relevant ads. You can delete the permission if necessary.

Learn more about the permission and ad ID

The ad ID is a unique identifier provided by Google Play services for displaying ads to users who opt in to personalized ads. Users can opt out of ad personalization or reset their ID in the settings. In this case, advertising networks won't be able to use the ID to select relevant ads for the user.

How to delete the permission

If some policies (such as [Google Play's Families Policy](#)) do not allow the use of ad IDs, you can delete the permission from the `AndroidManifest.xml` file.

```
<uses-permission android:name="com.google.android.gms.permission.AD_ID" tools:node="remove"/>
```

iOS

- In Podlist of your app's iOS module, add:

```
pod 'YandexMobileAdsMediation', '5.9.0'
```

- Mobile Ads SDK supports tracking of app installations using the [SKAdNetwork](#) framework. Installation tracking works for any device, even if no access to IDFA was granted.

To enable this functionality, add the IDs of the supported ad networks to the Info.plist file of your app's iOS module.

```
<key>SKAdNetworkItems</key>
<array>
  <dict>
    <!-- Yandex Ads -->
    <key>SKAdNetworkIdentifier</key>
    <string>zq4921623r.skadnetwork</string>
  </dict>
  <dict>
    <!-- AdMob -->
    <key>SKAdNetworkIdentifier</key>
    <string>cstr6suwn9.skadnetwork</string>
  </dict>
  <dict>
    <!-- MyTarget -->
    <key>SKAdNetworkIdentifier</key>
    <string>n9x2a789qt.skadnetwork</string>
  </dict>
  <dict>
    <!-- MyTarget -->
    <key>SKAdNetworkIdentifier</key>
    <string>r26jy69rp1.skadnetwork</string>
  </dict>
  <dict>
    <!-- Start.io -->
    <key>SKAdNetworkIdentifier</key>
    <string>5l3tpt7t6e.skadnetwork</string>
  </dict>
  <dict>
    <!-- UnityAds -->
    <key>SKAdNetworkIdentifier</key>
    <string>4dzt52r2t5.skadnetwork</string>
  </dict>
  <dict>
    <!-- IronSource -->
    <key>SKAdNetworkIdentifier</key>
    <string>su67r6k2v3.skadnetwork</string>
  </dict>
  <dict>
    <!-- Applovin -->
    <key>SKAdNetworkIdentifier</key>
    <string>ludvb6z3bs.skadnetwork</string>
  </dict>
  <dict>
    <!-- Mintegral -->
    <key>SKAdNetworkIdentifier</key>
    <string>KBD757YWX3.skadnetwork</string>
  </dict>
</array>
```

For more information, see [Configuring a Source App](#) in the Apple documentation.

Initializing the library

In the `initState` of your app's widget, add the line: `MobileAds.initialize();`

See also

[Enabling the plugin for YAN](#)

[COPPA](#)

[GDPR](#)

Ad formats

Enabling banner ads



Warning:

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

A *banner* is a configurable ad that covers part of the screen and reacts to clicks.

Enabling a banner

1. Add the import:

```
import 'package:yandex_mobileads/mobile_ads.dart';
```

2. Create an object of the `BannerAd` class in the `Builder` or `initState` of the widget hosting the banner:

```
final banner = BannerAd(  
  adUnitId: 'R-M-XXXXXX-Y',  
  // Flex-size  
  adSize: AdSize.flexible(width: screenWidth, height: bannerHeight),  
  // Sticky-size  
  adSize: AdSize.sticky(width: screenWidth),  
  adRequest: AdRequest(),  
  onAdLoaded: () {  
    /* Do something */  
  },  
  onAdFailedToLoad: (error) {  
    /* Do something */  
  },  
);
```

3. In the `build` method of your widget, specify the `AdWidget` that will accept the new `BannerAd` object:

```
AdWidget(bannerAd: banner)
```

4. The ad will load automatically. If necessary, you can call the `load` method of the banner object after you call `build` for the first time.

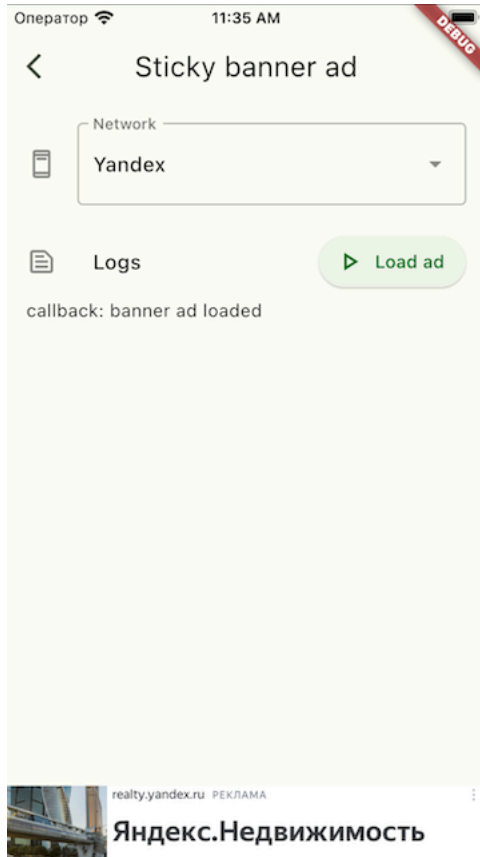
Example of working with banner ads

The code demonstrates how to create and configure `BannerAd` objects:

```
import 'package:yandex_mobileads/mobile_ads.dart';  
  
// ...  
  
class BannerAdPage extends StatelessWidget {  
  final banner = BannerAd(  
    adUnitId: 'demo-banner-yandex',  
    adSize: AdSize.sticky(width: screenWidth),  
    adRequest: AdRequest(),  
    onAdLoaded: () {  
      /* Do something */  
    },  
    onAdFailedToLoad: (error) {  
      /* Do something */  
    },  
  );  
};
```

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    body: Align(
      alignment: Alignment.bottomCenter,
      child: AdWidget(bannerAd: banner),
    ),
  );
}
```

If an ad is integrated this way, the banner appears after the app starts:



To see how the banner ad will be displayed in the app, use a demo AdUnitId:

- demo-banner-yandex

Enabling interstitial ads



Warning:

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

An *interstitial ad* is a configurable full-screen ad that reacts to clicks.

Creating and displaying InterstitialAd

1. Add the import:

```
import 'package:yandex_mobileleads/mobile_ads.dart';
```

2. Create an `InterstitialAd` class object. You can only use the asynchronous method to create the object:

```
final ad = await InterstitialAd.create(
  adUnitId: 'R-M-XXXXXX-Y',
  onAdLoaded: () {
    /* Do something */
  },
);
```

```

    onAdFailedToLoad: (error) {
      /* Do something */
    },
  );

```

3. After you create the `InterstitialAd` class object, load the ads. To load an ad, use the `load` method that accepts an optional `AdRequest` object.

```
await ad.load(adRequest: AdRequest());
```

You can wait for your ad to load asynchronously using the `await` keyword. If there's an error, the `onAdFailedToLoad` method is called to terminate waiting.

4. Interstitial ads are loaded in the background immediately after the `load` method is called. To display an interstitial ad, call the `show` method:

```
await ad.show();
```

You can wait for the ad to start displaying asynchronously.

5. Alternatively, you can use the `waitForDismiss` asynchronous method to wait until the end of ad serving:

```
await ad.waitForDismiss();
```

Example of working with interstitial ads

The code demonstrates how to create and configure an `InterstitialAd` object, and how to create and display an interstitial ad:

```

Future<void> showInterstitialAd() async {
  final ad = await InterstitialAd.create(
    adUnitId: 'demo-interstitial-yandex',
    onAdLoaded: () {
      /* Do something */
    },
    onAdFailedToLoad: (error) {
      /* Do something */
    },
  );
  await ad.load(adRequest: AdRequest());
  await ad.show();
  await ad.waitForDismiss();
}

```

If an ad is integrated this way, the ad unit appears when `showInterstitialAd` is called.

To see how the ad will be displayed in the app, use a demo `AdUnitId`:

- `demo-interstitial-yandex`

Enabling rewarded ads



Warning:

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

A *rewarded ad* is a configurable full-screen ad. The user gets a reward for viewing the ad.

Creating and displaying RewardedAd

1. Add the import:

```
import 'package:yandex_mobileleads/mobile_ads.dart';
```

2. Create a `RewardedAd` class object. You can only use the asynchronous method to create the object:

```

final ad = await RewardedAd.create(
  adUnitId: 'R-M-XXXXXX-Y',
  onAdLoaded: () {
    /* Do something */
  },
  onAdFailedToLoad: (error) {

```



```

        /* Do something */
    },
);

```

- After you create the RewardedAd class object, load the ads. To load an ad, use the load method that accepts an optional AdRequest object.

```
await ad.load(adRequest: AdRequest());
```

You can wait for your ad to load asynchronously using the await keyword.

- Rewarded ads are loaded in the background immediately after the load method is called. To display an ad, call the show method:

```
await ad.show();
```

You can wait for the ad to start displaying asynchronously.

- Alternatively, you can use the waitForDismiss asynchronous method to wait until the end of the ad serving and/or reward:

```
Reward? reward = await ad.waitForDismiss();
```

Example of working with rewarded ads

The code demonstrates how to create and configure a RewardedAd object, and how to load and display a rewarded ad:

```

Future<void> showRewardedAd() async {
  final ad = await RewardedAd.create(
    adUnitId: 'demo-rewarded-yandex',
    onAdLoaded: () {
      /* Do something */
    },
    onAdFailedToLoad: (error) {
      /* Do something */
    },
  );
  await ad.load(adRequest: AdRequest());
  await ad.show();
  final reward = await ad.waitForDismiss();
  if (reward != null) debugPrint('got ${reward.amount} of ${reward.type}');
}

```

If an ad is integrated this way, the ad unit appears when showRewardedAd is called.

To see how the ad will be displayed in the app, use a demo AdUnitId:

- demo-rewarded-yandex

Mobile mediation adapters



Warning:

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

Mobile mediation is a platform for automatically selecting ads from multiple ad networks. Each advertising network offers an ad to be displayed and the mediation platform chooses the most profitable one.

The mobile mediation platform integrates adapters from most of the major ad networks listed below.

List of supported ad networks

Ad network	Banner Ad	Interstitial Ad	Rewarded Ad
AdMob			

Ad network	Banner Ad	Interstitial Ad	Rewarded Ad
myTarget	✓	✓	✓
Start.io	✓	✓	✓
UnityAds	✗	✓	✓
AppLovin	✓	✓	✓
	(Android)		
IronSource	✗	✓	✓
AdColony (Android)	✓	✓	✓
ChartBoost (Android)	✓	✓	✓
Pangle (Android)	✗	✓	✓
Tapjoy (Android)	✗	✓	✓
Vungle (Android)	✓	✓	✓
Mintegral	✓	✓	✓

Enabling AdMob



Warning:

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

Supported ad formats

- [Banner ads](#)
- [Interstitial ads](#)
- [Rewarded ads](#)

Integration

To enable AdMob mediation, use the instructions for [Android](#) and [iOS](#).

Enabling myTarget



Warning:

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

Supported ad formats

- [Banner ads](#)
- [Interstitial ads](#)
- [Rewarded ads](#)

Integration

To enable myTarget mediation, use the instructions for [Android](#) and [iOS](#).

Enabling Start.io

**Warning:**

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

Supported ad formats

- [Banner ads](#)
- [Interstitial ads](#)
- [Rewarded ads](#)

Integration

To enable Start.io mediation, follow the instructions for [Android](#) and [iOS](#).

Enabling UnityAds

**Warning:**

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

Supported ad formats

- [Interstitial ads](#)
- [Rewarded ads](#)

Integration

To enable UnityAds mediation, use the instructions for [Android](#) and [iOS](#).

Enabling AppLovin

**Warning:**

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

Supported ad formats

- Android [banner ads](#)
- [Interstitial ads](#)
- [Rewarded ads](#)

Integration

To enable AppLovin mediation, use the instructions for [Android](#) and [iOS](#).

Enabling IronSource

**Warning:**

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

Supported ad formats

- [Interstitial ads](#)
- [Rewarded ads](#)

Integration

To enable ironSource mediation, follow the instructions for [Android](#) and [iOS](#).

Enabling AdColony

**Warning:**

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

Supported ad formats

- [Banner ads](#)
- [Interstitial ads](#)
- [Rewarded ads](#)

Integration

To enable AdColony mediation, use the instructions for [Android](#).

Enabling ChartBoost

**Warning:**

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

Supported ad formats

- [Banner ads](#)
- [Interstitial ads](#)
- [Rewarded ads](#)

Integration

To enable ChartBoost mediation, use the instructions for [Android](#).

Enabling Pangle

**Warning:**

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

Supported ad formats

- [Interstitial ads](#)
- [Rewarded ads](#)

Integration

To enable myTarget mediation, use the instructions for [Android](#).

Enabling Tapjoy

**Warning:**

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

Supported ad formats

- [Banner ads](#)
- [Interstitial ads](#)
- [Rewarded ads](#)

Integration

To enable tapJoy mediation, follow the instructions for [Android](#).

Enabling Vungle

**Warning:**

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

Supported ad formats

- [Banner ads](#)
- [Interstitial ads](#)
- [Rewarded ads](#)

Integration

To enable Vungle mediation, use the instructions for [Android](#).

Enabling Mintegral

**Warning:**

This is an archived version of the documentation. Actual documentation for all platforms can be found [here](#).

Supported ad formats

- [Banner ads](#)
- [Interstitial ads](#)
- [Rewarded ads](#)

Integration

To enable Mintegral mediation, use the instructions for [Android](#) and [iOS](#).