Freestar Flutter Plugin

Last Modified on 05/31/2022 12:28 pm EDT

Welcome to the freestar_flutter_plugin wiki!

The latest version of our plugin can be found here.

Add Freestar Ads Mediation to your Flutter app. Supported on Android and iOS platforms.

Run our sample

A great way to become familiar with Freestar mediation is to run our sample app.

1. If not already done so, install the Flutter development environment by Google!
2. Connect an Android device to your computer. (Don't worry, the plugin also supports iOS!)
3. Open command-shell or terminal window and:

```
git clone https://github.com/freestarcapital/freestar_flutter_plugin
```

```
cd freestar_flutter_plugin/example
```

```
flutter run
```

Getting Started

Follow these steps to integrate Freestar Flutter plugin (`freestar_flutter_plugin`) package into your Flutter app. The first step will be Dart setup. The Android and iOS sub-sections will follow.

Dart Setup

The following are essentially excerpts from our example main.dart Please take a look to see how easy it is to integrate Freestar mediation.

Initialize Freestar

```
import 'package:freestar_flutter_plugin/freestar_flutter_plugin.dart';
```

In your `initState` method, setup Freestar with the following test keys:

```
FreestarFlutterPlugin.enableLogging(true);        //set false for production
FreestarFlutterPlugin.enableTestMode(true);       //set false for production
FreestarFlutterPlugin.enablePartnerChooser(true); //set false for production

if (defaultTargetPlatform == TargetPlatform.android) {
  FreestarFlutterPlugin.init("XqjhRR"); //Android Test Key
} else if (defaultTargetPlatform == TargetPlatform.iOS) {
  FreestarFlutterPlugin.init("P8RIA3");  //iOS Test Key
}
```

Interstitial Ad (Fullscreen ad)

Make sure to implement the `InterstitialAdListener` interface. For example:

```
class _MyAppState extends State<MyApp>
    implements InterstitialAdListener
```

Declare/Instantiate `InterstitialAd`

```
InterstitialAd _interstitialAd = new InterstitialAd();

// Another way:
InterstitialAd _interstitialAd = InterstitialAd.from('optional-placement', myInterstitialAdListener);

// Loading
_interstitialAd.interstitialAdListener = myInterstitialAdListener; //if not already set, do so before calling loadAd
_interstitialAd.placement = 'optional-placement';  //optional placement or simply omit if not needed

//optional targeting parameters
Map targetingParams = Map();
targetingParams["my-targeting-param1"] = "example value 1";
targetingParams["my-targeting-param2"] = "example value 2";
_interstitialAd.targetingParams = targetingParams; //optional targeting params or simply omit if not needed

_interstitialAd.loadAd();  //fetches highest-paying mediation partner ad-fill
```

When you receive the `onInterstitialAdLoaded` callback, the interstitial ad is ready to be shown:

```
_interstitialAd.showAd();  //Display fullscreen interstitial ad
```

Rewarded Ad (Fullscreen ad)

Make sure to implement the `RewardedAdListener` interface. For example:

```
class _MyAppState extends State<MyApp>
    implements RewardedAdListener
```

Declare/Instantiate `RewardedAd`

```
RewardedAd _rewardedAd = new RewardedAd();

// Another way:
RewardedAd _rewardedAd = RewardedAd.from('optional-placement', myRewardedAdListener);

// Loading
_rewardedAd.rewardedAdListener = myRewardedAdListener; //if not already set, do so before calling loadAd
_rewardedAd.placement = 'optional-placement';  //optional placement or simply omit if not needed

//optional targeting parameters
Map targetingParams = Map();
targetingParams["my-targeting-param1"] = "example value 1";
targetingParams["my-targeting-param2"] = "example value 2";
_rewardedAd.targetingParams = targetingParams; //optional targeting params or simply omit if not needed

_rewardedAd.loadAd();  //fetches highest-paying mediation partner ad-fill
```

When you receive the `onRewardedAdLoaded` callback, the Rewarded ad is ready to be shown:

```
//secret - the secret string you want to secure your reward with
//userId - your user id string.  can use empty string ""
//rewardType - can be "Gold" or "Coins" or whatever your virtual currency may be!
//rewardAmount - whatever string amount your user will be rewarded with

_rewardedAd.showAd("my-secret-12345", "myUserId-12345", "V-Bucks", "100");  //Display fullscreen Rewarded ad
```

When 'onRewardedAdCompleted' is called, this indicates the user has completed viewing the fullscreen rewarded to the end.

Banner Ad

The following banner ad sizes are supported:

```
BannerAd.AD_SIZE_BANNER_320x50
BannerAd.AD_SIZE_MREC_300x250
BannerAd.AD_SIZE_LEADERBOARD_728x90   (for tablet devices)
```

Make sure to implement the `BannerAdListener` interface. For example:

```
class _MyAppState extends State<MyApp>
    implements BannerAdListener
```

Declare/Instantiate `BannerAd`

```
BannerAd _bannerAd = new BannerAd();

// Another way:
BannerAd _bannerAd = BannerAd.from('optional-placement', BannerAd.AD_SIZE_BANNER_320x50, myRewardedAdLis
tener, false);

//The last parameter is `doAutoloadWhenCreated` (see below for reference)
```

Add BannerAd to your UI

Add the BannerAd to your Scaffold or UI hierarchy:

```
_bannerAd.bannerAdListener = myBannerAdListener;    //set listener, if not already set
_bannerAd.adSize = BannerAd.AD_SIZE_BANNER_320x50;  //set the banner ad size, if not already not
_bannerAd.doAutoloadWhenCreated = true;  //displays the banner ad as soon as UI is initialized and do not need to ca
ll bannerAd.loadAd

//Small banner ad example:
Container(width: 320.0,  //explicitly set the physical container size appropriately
        height: 50.0,
        child: _bannerAd)

//MREC banner ad example:
Container(width: 300.0,  //explicitly set the physical container size appropriately
        height: 250.0,
        child: _mrecBannerAd)
```

Banner ad also supports optional targeting parameters:

```
Map targetingParams = Map();
targetingParams["my-targeting-param1"] = "example value 1";
targetingParams["my-targeting-param2"] = "example value 2";
_bannerAd.targetingParams = targetingParams; //optional targeting params or simply omit if not needed

_bannerAd.loadAd();  //fetches highest-paying mediation partner ad-fill
```

When the a fill is received, the BannerAd will automatically display in the position where it was added to the UI and `onBannerAdLoaded` will be called.

Native Ad

Native Ads are supported. Out of the box, native ads support two general sizes:

```
NativeAd.NATIVE_TEMPLATE_SMALL    AND
NativeAd.NATIVE_TEMPLATE_MEDIUM
```

Make sure to implement the `NativeAdListener` interface. For example:

```
class _MyAppState extends State<MyApp>
    implements NativeAdListener
```

Declare/Instantiate `NativeAd`

```
NativeAd _nativeAd = new NativeAd();

// Another way:
NativeAd _nativeAd = NativeAd.from('optional-placement', NativeAd.NATIVE_TEMPLATE_SMALL, myNativeAdListener, false);

//The last parameter is `doAutoloadWhenCreated` (see below for reference)
```

Add NativeAd to your UI

Add the NativeAd to your Scaffold or UI hierarchy:

```
_nativeAd.nativeAdListener = myNativeAdListener;    //set listener, if not already set
_nativeAd.adSize = NativeAd.NATIVE_TEMPLATE_SMALL;  //set the native ad size, if not already not
_nativeAd.doAutoloadWhenCreated = true;  //displays the native ad as soon as UI is initialized and do not need to call nativeAd.loadAd

//Small template native ad example:
Container(width: window.physicalSize.width,  //fullscreen width
        height: 100.0,  //100.0 is the small native ad height
        child: _smallNativeAd)

//Medium template native ad example:
Container(width: window.physicalSize.width,  //fullscreen width
        height: 350.0,  //350.0 is the medium native ad height
        child: _mediumNativeAd)
```

Native ad also supports optional targeting parameters:

```
Map targetingParams = Map();
targetingParams["my-targeting-param1"] = "example value 1";
targetingParams["my-targeting-param2"] = "example value 2";
_nativeAd.targetingParams = targetingParams; //optional targeting params or simply omit if not needed

_nativeAd.loadAd();  //fetches highest-paying mediation partner ad-fill
```

When the a fill is received, the NativeAd will automatically display in the position where it was added to the UI and `onNativeAdLoaded` will be called.

Android Setup

Add this section to your app AndroidManifest.xml. More specifically, just the `meta-data` tags. Using our example, the path would be: `freestar_flutter_plugin/example/android/app/src/main/AndroidManifest.xml`

When done, you may run your Flutter app on a connected Android device!

Android Proguard

When creating a release version of your Android app, make sure to use our proguard-rules.pro from our flutter sample ads app.

Android plugin dependencies

See our plugin gradle dependencies here

iOS Setup

The `flutter build ios` command in the terminal will generate an Xcode project from your Flutter app. The generated folder (`example/ios`) in this repo will include:

- An `.xcworkspace` file
- An `.xcproject` file
- A `{YOUR_APP_NAME}` folder containing generated native code
- A `Podfile` file
- A `Podfile.lock` file
- A `Pods` folder

The last three relate to the CocoaPods system, used both by Flutter and by Freestar to manage iOS library dependencies. To complete the setup, you will need to edit the `Podfile` with the following changes:

1. Specify the iOS version: `platform :ios, '10.0'` . This is the minimum supported version. Freestar supports iOS 10.0 and above.
2. In the `target '{YOUR_APP_NAME}'` section, list the advertsing partners from which you would like to see ads:

```
pod 'FreestarAds-AdColony', '~> 5.0'
pod 'FreestarAds-AppLovin', '~> 5.0'
pod 'FreestarAds-Googleadmob', '~> 6.0'
pod 'FreestarAds-Tapjoy', '~> 5.0'
pod 'FreestarAds-Unity', '~> 6.0'
pod 'FreestarAds-Vungle', '~> 5.0'
pod 'FreestarAds-Criteo', '~> 3.0'
pod 'FreestarAds-GAM', '~> 5.0'
pod 'FreestarAds-Mopub', '~> 5.0'
pod 'FreestarAds-Nimbus', '~> 3.0'
pod 'FreestarAds-TAM', '~> 2.0'
pod 'FreestarAds-Pangle', '~> 1.0'
pod 'FreestarAds-Hyprmx', '~> 1.0'
```

Once you made the changes, save the `Podfile` , enter the `ios` directory in the terminal, and re-

run `pod install`. Then open the `{YOUR_APP_NAME}.xcworkspace` file with Xcode.

The generated project will include an `Info.plist` file. If, when editing your `Podfile` (see above), you included either `FreestarAds-Googleadmob` or `FreestarAds-GAM` in the partner list, you will need to add your Google Application Identifier (received from Google) to the `Info.plist`; the app will not function without this:

```
<key>GADApplicationIdentifier</key>
<string>ca-app-pub-3940256099942544~1458002511</string>
```

There are also two flags that you can add to the plist to aid in development. To see test ads:

```
<key>CHP_TEST_ADS</key>
<string>enable</string>
```

Test ads do not generate revenue, remove this flag before submitting the app to the App Store.

To generate detailed logging from the Freestar SDK:

```
<key>CHP_LOGGING_ENABLE</key>
<string>true</string>
```

> ⚠Warning: For both performance and security reasons, it is not advisable to have detailed logging in production apps. Remove the flag in the Info.plist before submitting to the App Store.

When done, you may run your Flutter app on a connected iOS device!

Summary

Integrating Freestar Flutter plugin to display ads in your Flutter app is easy. As mentioned, please see our example main.dart