Lab session 3: Multi-Screens Design using Flutter

Task-3 Adding AdMob ads to a Flutter app (Part I)

This documentation provides a concise guide on integrating AdMob ads into the "Awesome Drawing Quiz" app. It accompanies a code lab that offers step-by-step instructions for the implementation process.

The "Awesome Drawing Quiz" app is an interactive game that challenges players to identify drawings and guess their corresponding names. By incorporating AdMob ads, we can monetize the app and generate revenue through mobile advertising.

This documentation covers the integration of three types of ads: banner ads, interstitial ads, and rewarded ads. Banner ads are small, persistent ads that can be placed at the top or bottom of the screen. Interstitial ads are full-screen ads that appear at natural transition points within the app. Rewarded ads provide incentives to users in exchange for their engagement with ads.

We will learn how to seamlessly integrate AdMob into Flutter apps, create ad units, load and display ads, handle ad events, and optimize ad performance.

Set up your Flutter development environment

To set up the Flutter development environment for this codelab, we will need to download two essential components: the Flutter SDK and an editor.

Ensure to have one of the following devices available for running the codelab:

- A physical Android or iOS device connected to your computer and set to Developer mode.
- The iOS simulator (requires installing Xcode tools).
- The Android Emulator (requires setup in Android Studio).
- A browser (Chrome is required for debugging).

Once we have the necessary device or platform set up, we can proceed to download the code for the codelab.

Set up the AdMob app and ad units

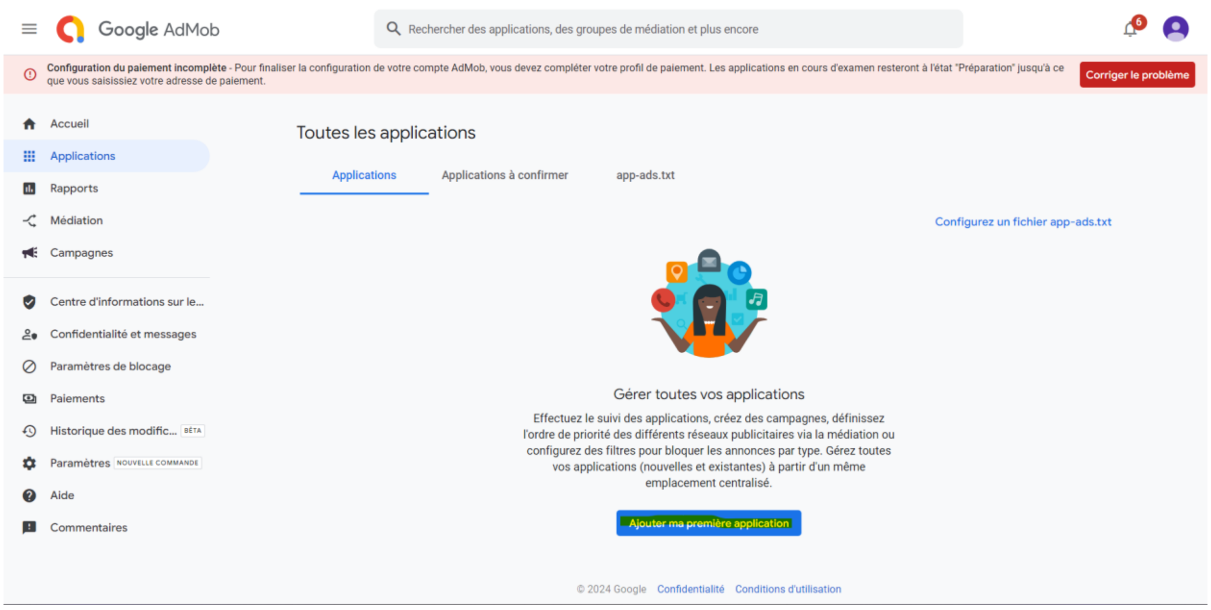
Because Flutter is a multi-platform SDK, we can add an app and ad units for both Android and iOS in AdMob.

Set up for Android

To set up for Android, you need to add an Android app and create ad units.

Add an Android app

Step 1: In the AdMob console, click ADD APP from the Apps menu.


Google AdMob
Rechercher des applications, des groupes de médiation et plus encore
Corriger le problème
Accueil
Applications
Rapports
Médiation
Campagnes
Centre d'informations sur le...
Confidentialité et messages
Paramètres de blocage
Paiements
Historique des modific... BÊTA
Paramètres NOUVELLE COMMANDE
Aide
Commentaires
Applications
Applications à confirmer
app-ads.txt
Configurez un fichier app-ads.txt
Ajouter ma première application
© 2024 Google Confidentialité Conditions d'utilisation

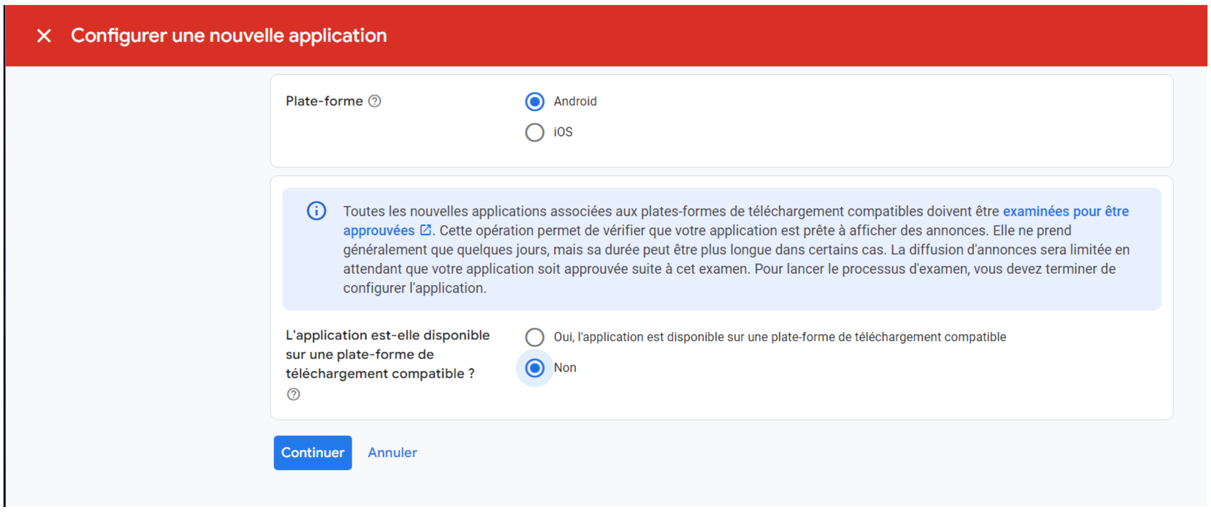
Configuration du paiement incomplète - Pour finaliser la configuration de votre compte AdMob, vous devez compléter votre profil de paiement. Les applications en cours d'examen resteront à l'état "Préparation" jusqu'à ce que vous saisissiez votre adresse de paiement.

Toutes les applications

Gérer toutes vos applications

Effectuez le suivi des applications, créez des campagnes, définissez l'ordre de priorité des différents réseaux publicitaires via la médiation ou configurez des filtres pour bloquer les annonces par type. Gérez toutes vos applications (nouvelles et existantes) à partir d'un même emplacement centralisé.

Step 2: When you're asked Have you published your app on Google Play or the App Store?, click NO.



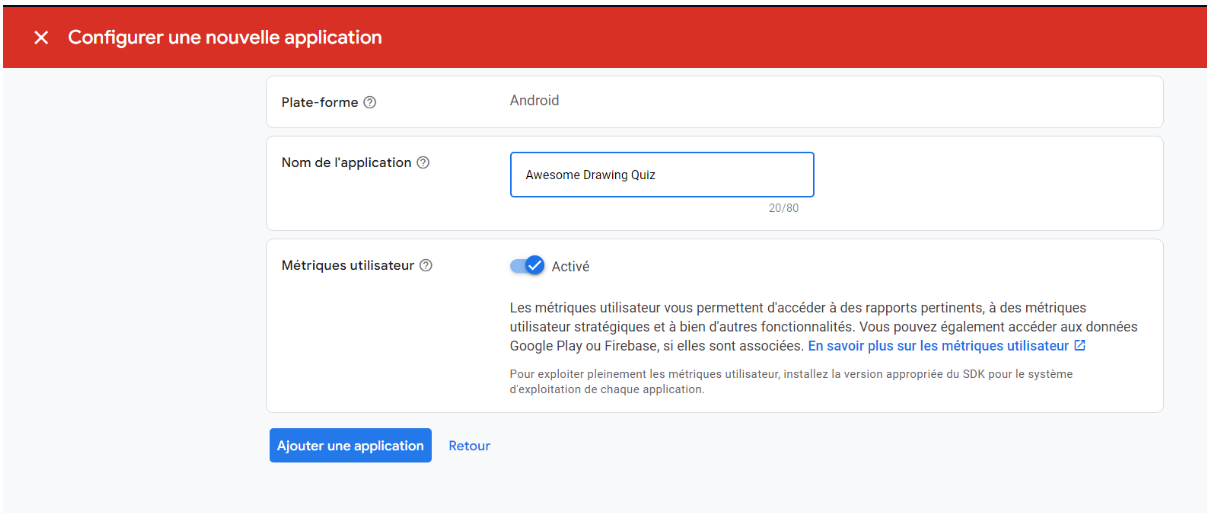
Configurer une nouvelle application

Plate-forme: [✓ Android] iOS

Toutes les nouvelles applications associées aux plates-formes de téléchargement compatibles doivent être examinées pour être approuvées. Cette opération permet de vérifier que votre application est prête à afficher des annonces. Elle ne prend généralement que quelques jours, mais sa durée peut être plus longue dans certains cas. La diffusion d'annonces sera limitée en attendant que votre application soit approuvée suite à cet examen. Pour lancer le processus d'examen, vous devez terminer de configurer l'application.

L'application est-elle disponible sur une plate-forme de téléchargement compatible ? Oui, l'application est disponible sur une plate-forme de téléchargement compatible [✓ Non]

Step 3: Enter Awesome Drawing Quiz in the app name field, and select Android as the platform.



Configurer une nouvelle application

Plate-forme: Android

Nom de l'application: Awesome Drawing Quiz 20/80

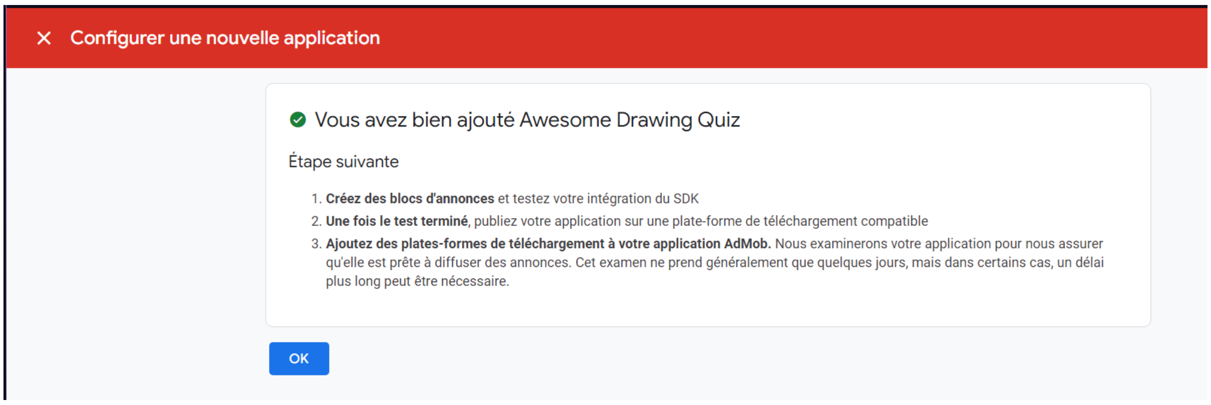
Métriques utilisateur: [✓ Activé]

Les métriques utilisateur vous permettent d'accéder à des rapports pertinents, à des métriques utilisateur stratégiques et à bien d'autres fonctionnalités. Vous pouvez également accéder aux données Google Play ou Firebase, si elles sont associées. En savoir plus sur les métriques utilisateur

Pour exploiter pleinement les métriques utilisateur, installez la version appropriée du SDK pour le système d'exploitation de chaque application.

Enabling user metrics is not necessary to complete this codelab. However, we recommend to do so because it allows you to understand user behavior in more detail.

Step 4: Click ADD to complete the process.



Configurer une nouvelle application

Vous avez bien ajouté Awesome Drawing Quiz

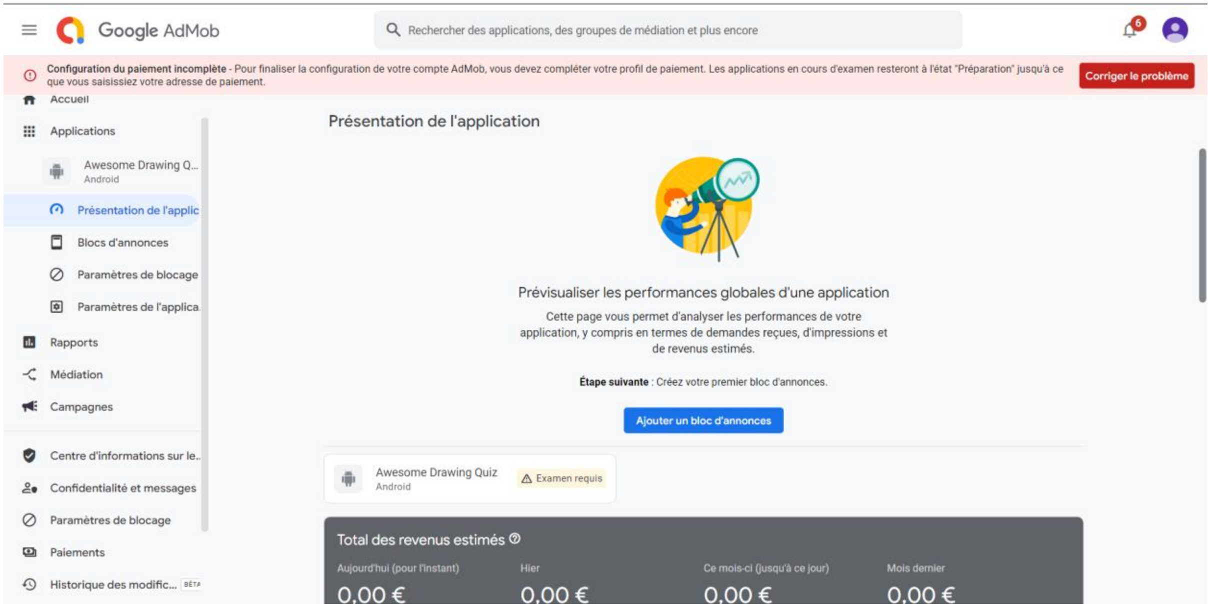
Étape suivante

1. **Créez des blocs d'annonces** et testez votre intégration du SDK
2. **Une fois le test terminé**, publiez votre application sur une plate-forme de téléchargement compatible
3. **Ajoutez des plates-formes de téléchargement à votre application AdMob.** Nous examinerons votre application pour nous assurer qu'elle est prête à diffuser des annonces. Cet examen ne prend généralement que quelques jours, mais dans certains cas, un délai plus long peut être nécessaire.

OK

Create ad units

Step 1: Select Awesome Drawing Quiz from the Apps menu in the AdMob console.



Google AdMob

Rechercher des applications, des groupes de médiation et plus encore

Configuration du paiement incomplète - Pour finaliser la configuration de votre compte AdMob, vous devez compléter votre profil de paiement. Les applications en cours d'examen resteront à l'état "Préparation" jusqu'à ce que vous saisissiez votre adresse de paiement.

Corriger le problème

Accueil
Applications
Awesome Drawing Q...
Android
Présentation de l'applic
Blocs d'annonces
Paramètres de blocage
Paramètres de l'applica
Rapports
Médiation
Campagnes
Centre d'informations sur le..
Confidentialité et messages
Paramètres de blocage
Paiements
Historique des modific... BÊTA

Présentation de l'application

Prévisualiser les performances globales d'une application

Cette page vous permet d'analyser les performances de votre application, y compris en termes de demandes reçues, d'impressions et de revenus estimés.

Étape suivante : Créez votre premier bloc d'annonces.

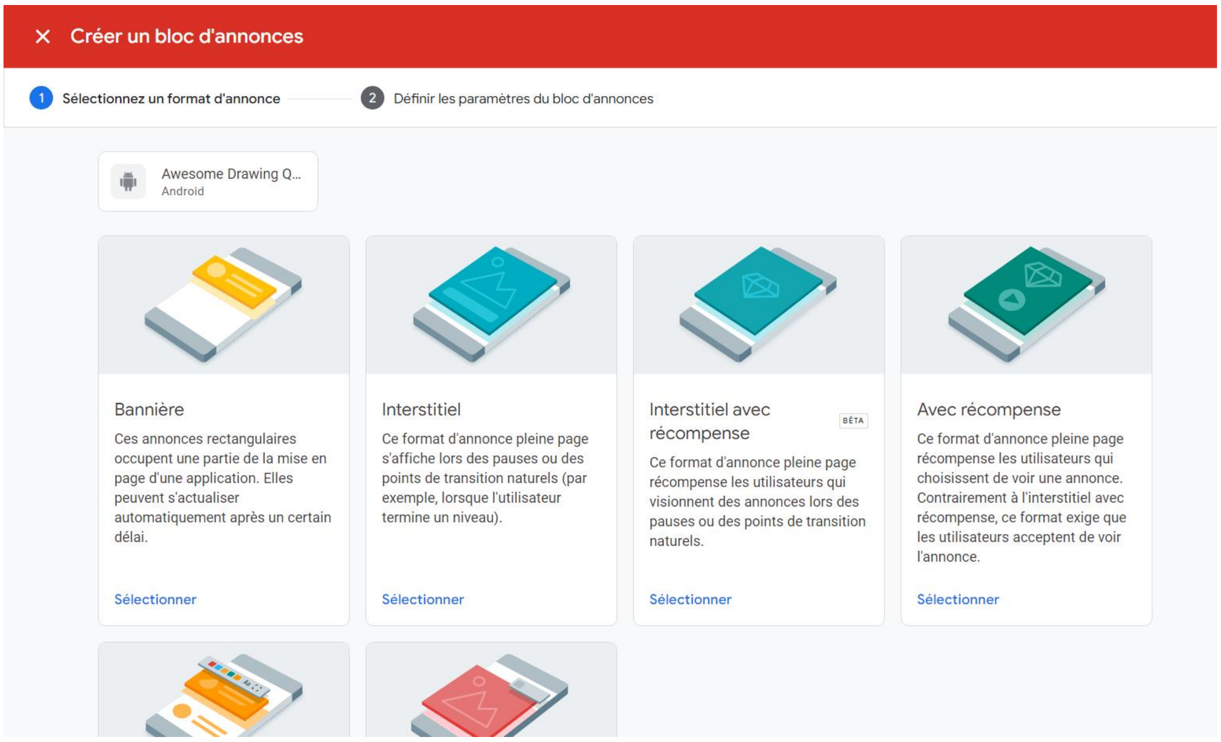
Ajouter un bloc d'annonces

Awesome Drawing Quiz
Android
Examen requis

Total des revenus estimés

Aujourd'hui (pour l'instant)	Hier	Ce mois-ci (jusqu'à ce jour)	Mois dernier
0,00 €	0,00 €	0,00 €	0,00 €

Step 2: Click the Ad units menu.


Créer un bloc d'annonces

1 Sélectionnez un format d'annonce — 2 Définir les paramètres du bloc d'annonces

Awesome Drawing Q...
Android

Bannière
Ces annonces rectangulaires occupent une partie de la mise en page d'une application. Elles peuvent s'actualiser automatiquement après un certain délai.
Sélectionner

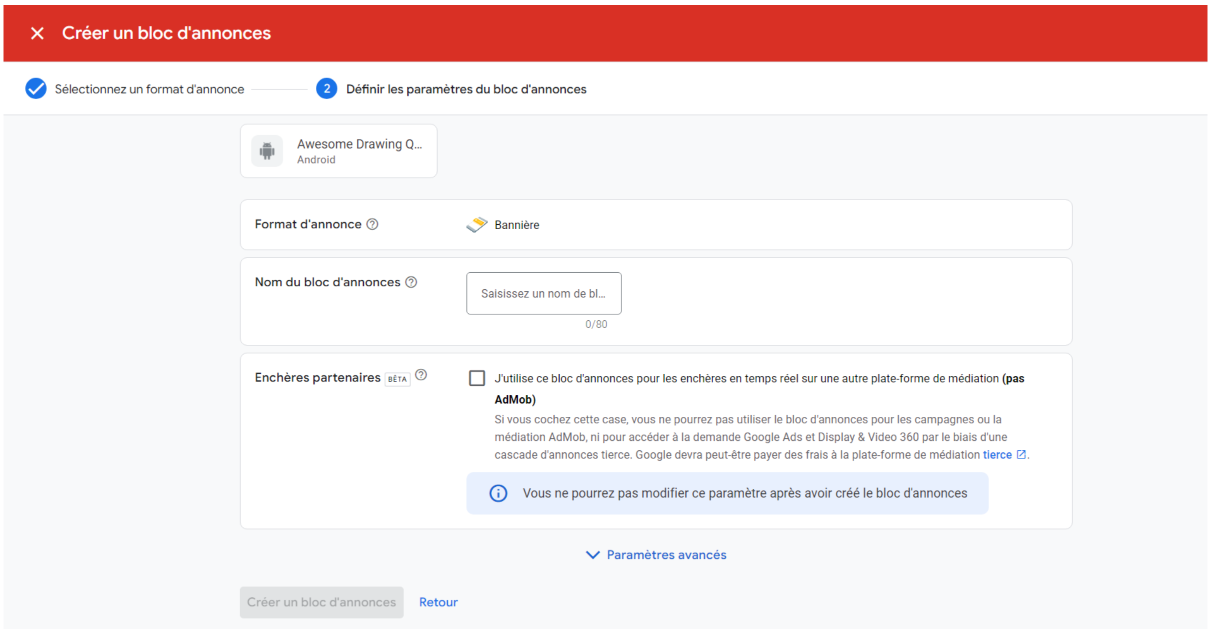
Interstitiel
Ce format d'annonce pleine page s'affiche lors des pauses ou des points de transition naturels (par exemple, lorsque l'utilisateur termine un niveau).
Sélectionner

Interstitiel avec récompense BÊTA
Ce format d'annonce pleine page récompense les utilisateurs qui visionnent des annonces lors des pauses ou des points de transition naturels.
Sélectionner

Avec récompense
Ce format d'annonce pleine page récompense les utilisateurs qui choisissent de voir une annonce. Contrairement à l'interstitiel avec récompense, ce format exige que les utilisateurs acceptent de voir l'annonce.
Sélectionner

Banner

1. Select Banner as the format.

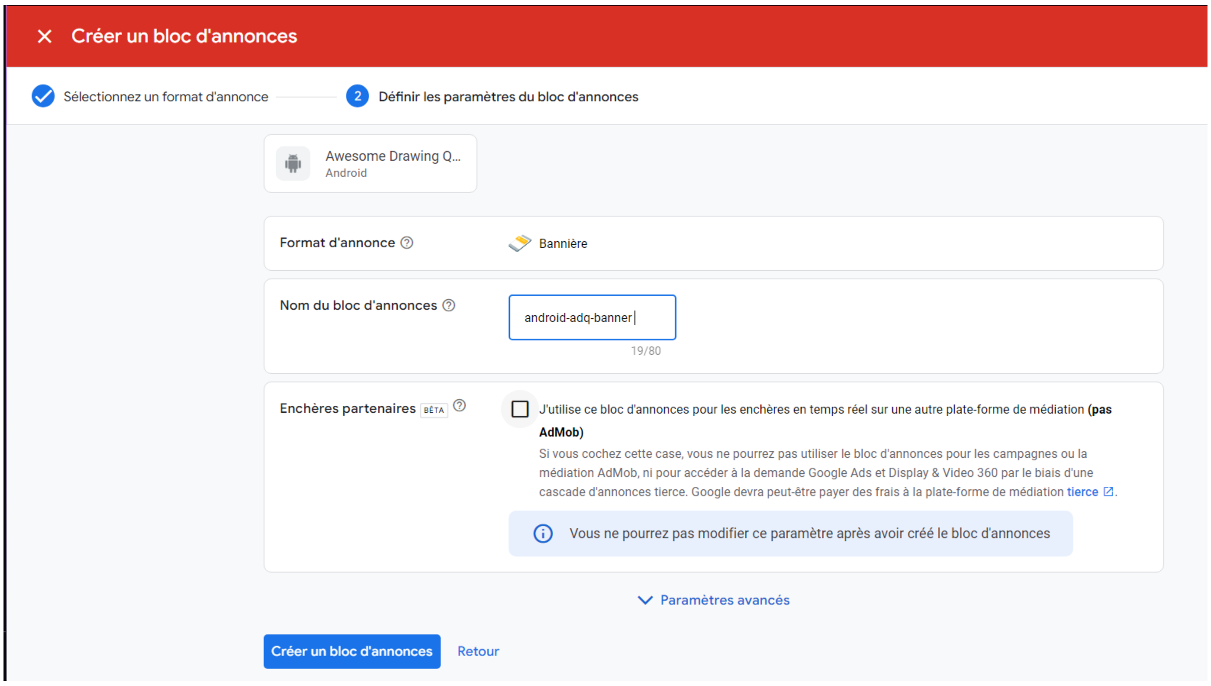

Créer un bloc d'annonces

✓ Sélectionnez un format d'annonce — 2 Définir les paramètres du bloc d'annonces

Awesome Drawing Q...
Android

Format d'annonce	Bannière
Nom du bloc d'annonces	Saisissez un nom de bl... 0/80
Enchères partenaires BÊTA	☐ J'utilise ce bloc d'annonces pour les enchères en temps réel sur une autre plate-forme de médiation **(pas AdMob)** Si vous cochez cette case, vous ne pourrez pas utiliser le bloc d'annonces pour les campagnes ou la médiation AdMob, ni pour accéder à la demande Google Ads et Display & Video 360 par le biais d'une cascade d'annonces tierce. Google devra peut-être payer des frais à la plate-forme de médiation tierce. Vous ne pourrez pas modifier ce paramètre après avoir créé le bloc d'annonces

2. Enter android-adq-banner in the Ad unit name field.



Créer un bloc d'annonces
Sélectionnez un format d'annonce
Définir les paramètres du bloc d'annonces
Awesome Drawing Q...
Android

Format d'annonce: Bannière

Nom du bloc d'annonces: android-adq-banner

19/80

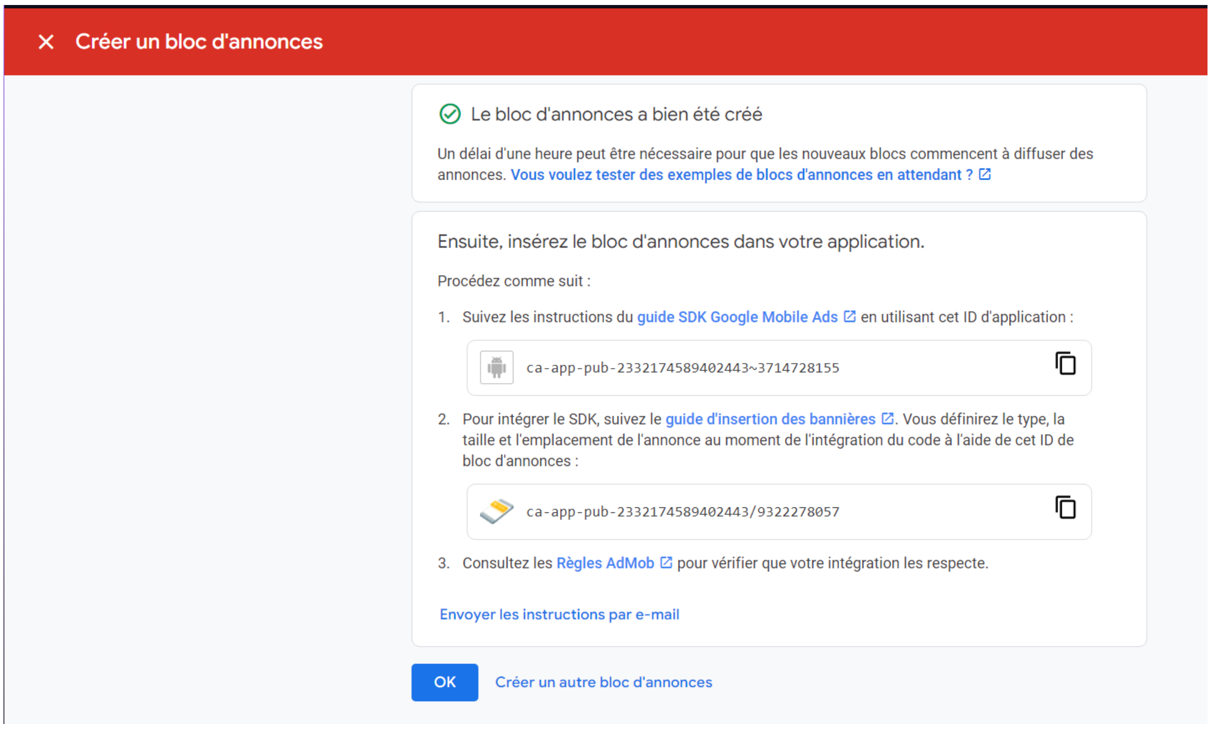
Enchères partenaires BÊTA

☐ J'utilise ce bloc d'annonces pour les enchères en temps réel sur une autre plate-forme de médiation **(pas AdMob)**

Si vous cochez cette case, vous ne pourrez pas utiliser le bloc d'annonces pour les campagnes ou la médiation AdMob, ni pour accéder à la demande Google Ads et Display & Video 360 par le biais d'une cascade d'annonces tierce. Google devra peut-être payer des frais à la plate-forme de médiation tierce.

Vous ne pourrez pas modifier ce paramètre après avoir créé le bloc d'annonces

3. Click CREATE AD UNIT to complete the process.



Le bloc d'annonces a bien été créé

Un délai d'une heure peut être nécessaire pour que les nouveaux blocs commencent à diffuser des annonces. Vous voulez tester des exemples de blocs d'annonces en attendant ?

Ensuite, insérez le bloc d'annonces dans votre application.

Procédez comme suit :

1. Suivez les instructions du guide SDK Google Mobile Ads en utilisant cet ID d'application :

```
ca-app-pub-2332174589402443~3714728155
```

2. Pour intégrer le SDK, suivez le guide d'insertion des bannières. Vous définirez le type, la taille et l'emplacement de l'annonce au moment de l'intégration du code à l'aide de cet ID de bloc d'annonces :

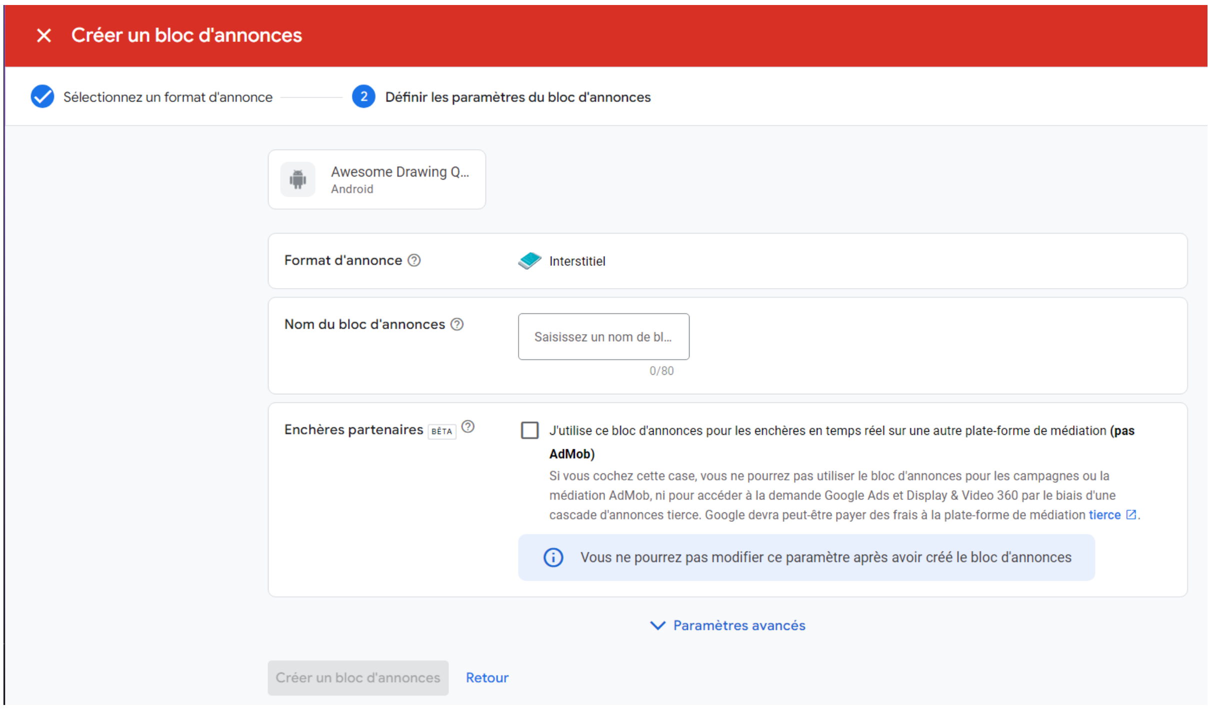
```
ca-app-pub-2332174589402443/9322278057
```

3. Consultez les Règles AdMob pour vérifier que votre intégration les respecte.

Envoyer les instructions par e-mail
OK
Créer un autre bloc d'annonces

Interstitial

1. Select Interstitial as the format.



Créer un bloc d'annonces

Sélectionnez un format d'annonce — 2 Définir les paramètres du bloc d'annonces

Awesome Drawing Q...
Android

Format d'annonce: Interstitiel

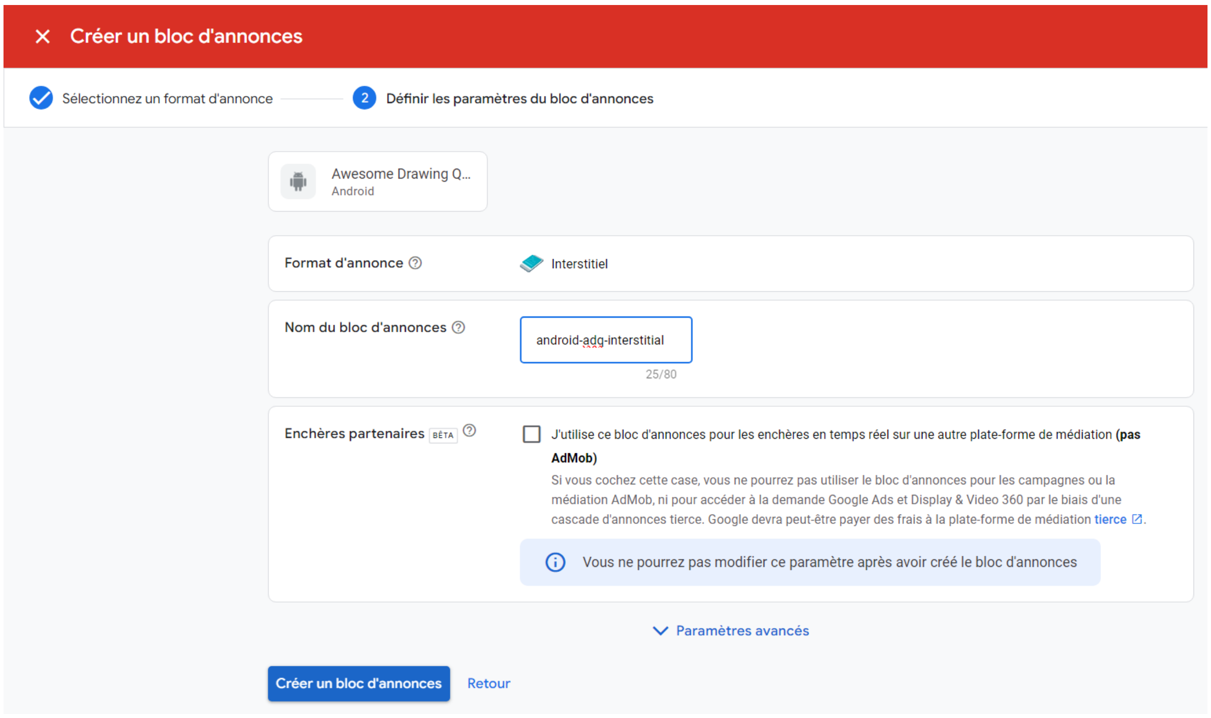
Nom du bloc d'annonces: Saisissez un nom de bl... 0/80

Enchères partenaires BÊTA: ☐ J'utilise ce bloc d'annonces pour les enchères en temps réel sur une autre plate-forme de médiation **(pas AdMob)**

Si vous cochez cette case, vous ne pourrez pas utiliser le bloc d'annonces pour les campagnes ou la médiation AdMob, ni pour accéder à la demande Google Ads et Display & Video 360 par le biais d'une cascade d'annonces tierce. Google devra peut-être payer des frais à la plate-forme de médiation tierce.

Vous ne pourrez pas modifier ce paramètre après avoir créé le bloc d'annonces

2. Enter android-adq-interstitial in the Ad unit name field.



Créer un bloc d'annonces

Sélectionnez un format d'annonce — 2 Définir les paramètres du bloc d'annonces

Awesome Drawing Q...
Android

Format d'annonce: Interstitiel

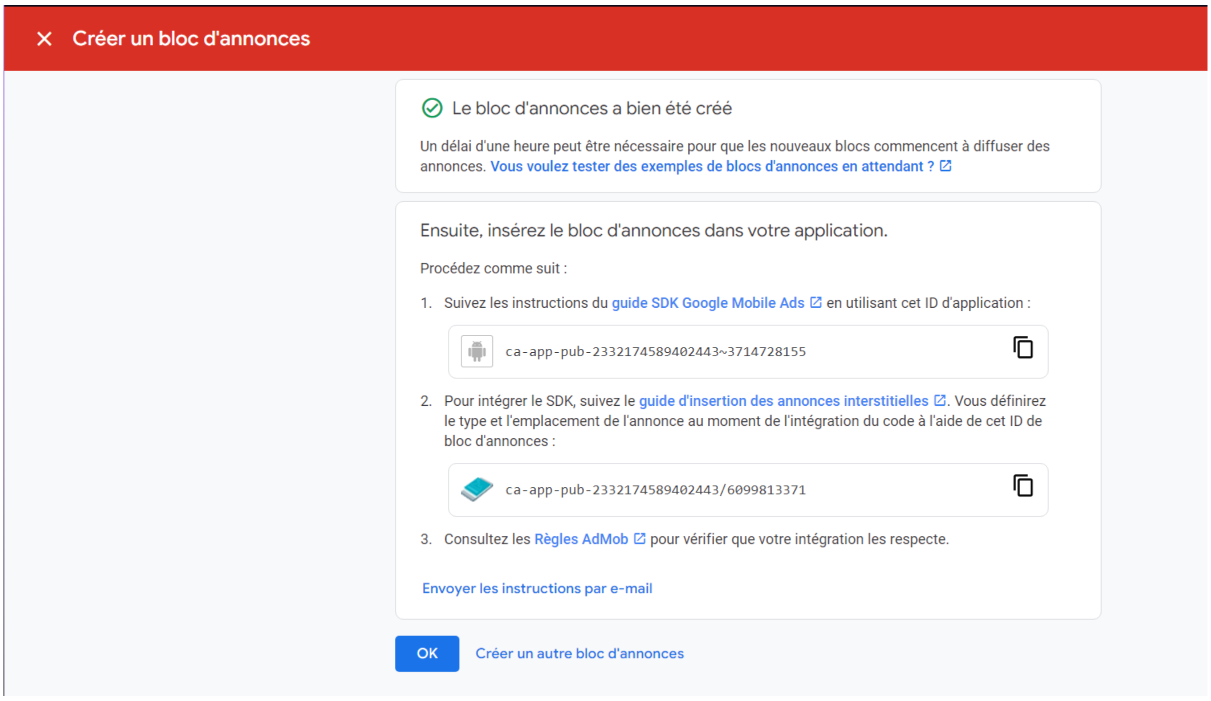
Nom du bloc d'annonces: android-adq-interstitial 25/80

Enchères partenaires BÊTA: ☐ J'utilise ce bloc d'annonces pour les enchères en temps réel sur une autre plate-forme de médiation **(pas AdMob)**

Si vous cochez cette case, vous ne pourrez pas utiliser le bloc d'annonces pour les campagnes ou la médiation AdMob, ni pour accéder à la demande Google Ads et Display & Video 360 par le biais d'une cascade d'annonces tierce. Google devra peut-être payer des frais à la plate-forme de médiation tierce.

Vous ne pourrez pas modifier ce paramètre après avoir créé le bloc d'annonces

3. Click CREATE AD UNIT to complete the process.



Créer un bloc d'annonces

Le bloc d'annonces a bien été créé

Un délai d'une heure peut être nécessaire pour que les nouveaux blocs commencent à diffuser des annonces. Vous voulez tester des exemples de blocs d'annonces en attendant ?

Ensuite, insérez le bloc d'annonces dans votre application.

Procédez comme suit :

1. Suivez les instructions du guide SDK Google Mobile Ads en utilisant cet ID d'application :

```
ca-app-pub-2332174589402443~3714728155
```

2. Pour intégrer le SDK, suivez le guide d'insertion des annonces interstitielles. Vous définirez le type et l'emplacement de l'annonce au moment de l'intégration du code à l'aide de cet ID de bloc d'annonces :

```
ca-app-pub-2332174589402443/6099813371
```

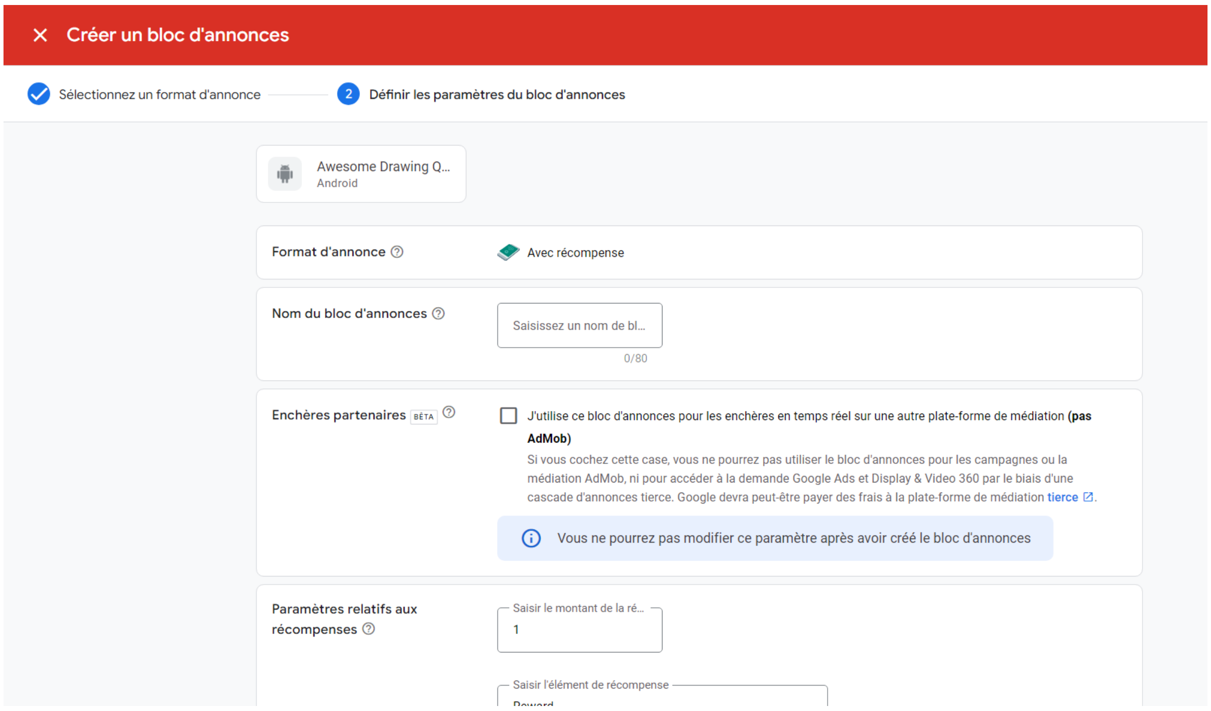
3. Consultez les Règles AdMob pour vérifier que votre intégration les respecte.

Envoyer les instructions par e-mail

OK | Créer un autre bloc d'annonces

Rewarded

1. Select Rewarded as the format.



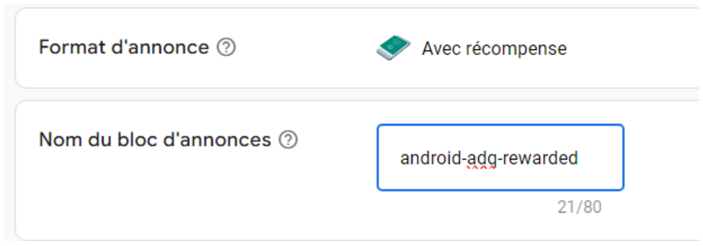
Créer un bloc d'annonces

Sélectionnez un format d'annonce — 2 Définir les paramètres du bloc d'annonces

Awesome Drawing Q...
Android

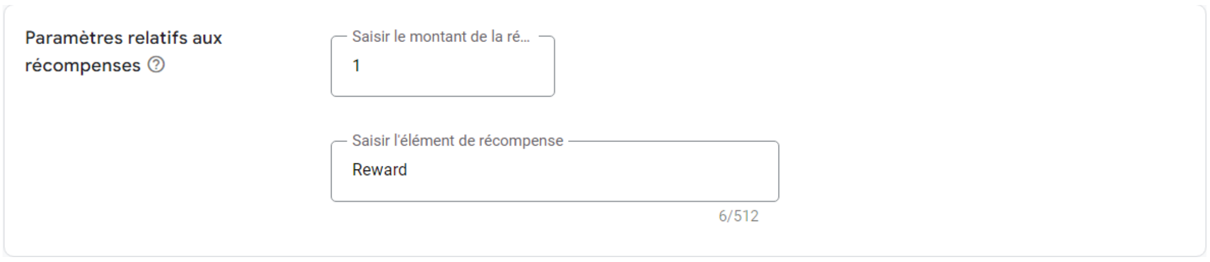
Format d'annonce	Avec récompense
Nom du bloc d'annonces	Saisissez un nom de bl... 0/80
Enchères partenaires BÊTA	J'utilise ce bloc d'annonces pour les enchères en temps réel sur une autre plate-forme de médiation (**pas AdMob**) Si vous cochez cette case, vous ne pourrez pas utiliser le bloc d'annonces pour les campagnes ou la médiation AdMob, ni pour accéder à la demande Google Ads et Display & Video 360 par le biais d'une cascade d'annonces tierce. Google devra peut-être payer des frais à la plate-forme de médiation tierce. Vous ne pourrez pas modifier ce paramètre après avoir créé le bloc d'annonces
Paramètres relatifs aux récompenses	Saisir le montant de la ré...: 1 Saisir l'élément de récompense

2. Enter android-adq-rewarded in the Ad unit name field.



Format d'annonce	Avec récompense
Nom du bloc d'annonces	android-adq-rewarded 21/80

3. Leave the default for Reward settings.

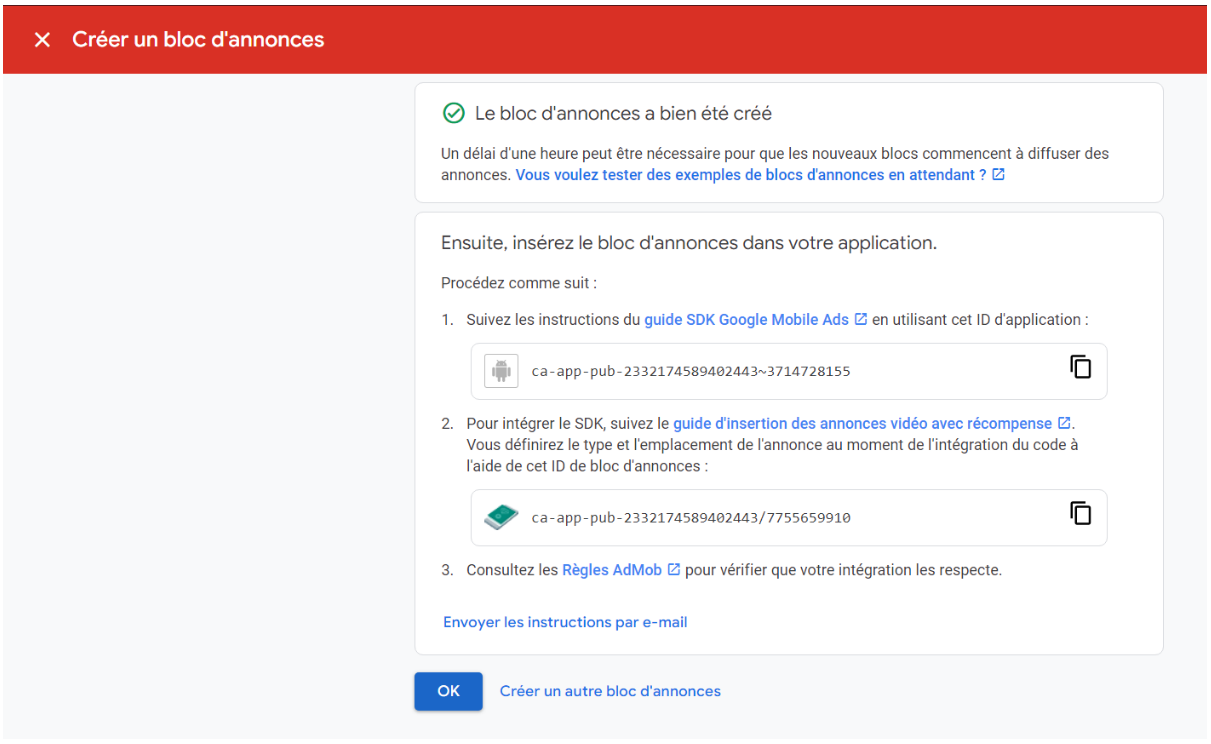


Paramètres relatifs aux récompenses

Saisir le montant de la ré...
1

Saisir l'élément de récompense
Reward
6/512

4. Click CREATE AD UNIT to complete the process.



Créer un bloc d'annonces

Le bloc d'annonces a bien été créé

Un délai d'une heure peut être nécessaire pour que les nouveaux blocs commencent à diffuser des annonces. Vous voulez tester des exemples de blocs d'annonces en attendant ?

Ensuite, insérez le bloc d'annonces dans votre application.

Procédez comme suit :

1. Suivez les instructions du guide SDK Google Mobile Ads en utilisant cet ID d'application :

   `ca-app-pub-2332174589402443~3714728155`

2. Pour intégrer le SDK, suivez le guide d'insertion des annonces vidéo avec récompense. Vous définirez le type et l'emplacement de l'annonce au moment de l'intégration du code à l'aide de cet ID de bloc d'annonces :

   `ca-app-pub-2332174589402443/7755659910`

3. Consultez les Règles AdMob pour vérifier que votre intégration les respecte.

Envoyer les instructions par e-mail

Set up for iOS

To set up for iOS, we need to add an iOS app and create ad units. To do so, we just follow the same process as for android but use ios-adq-… for ad units name.

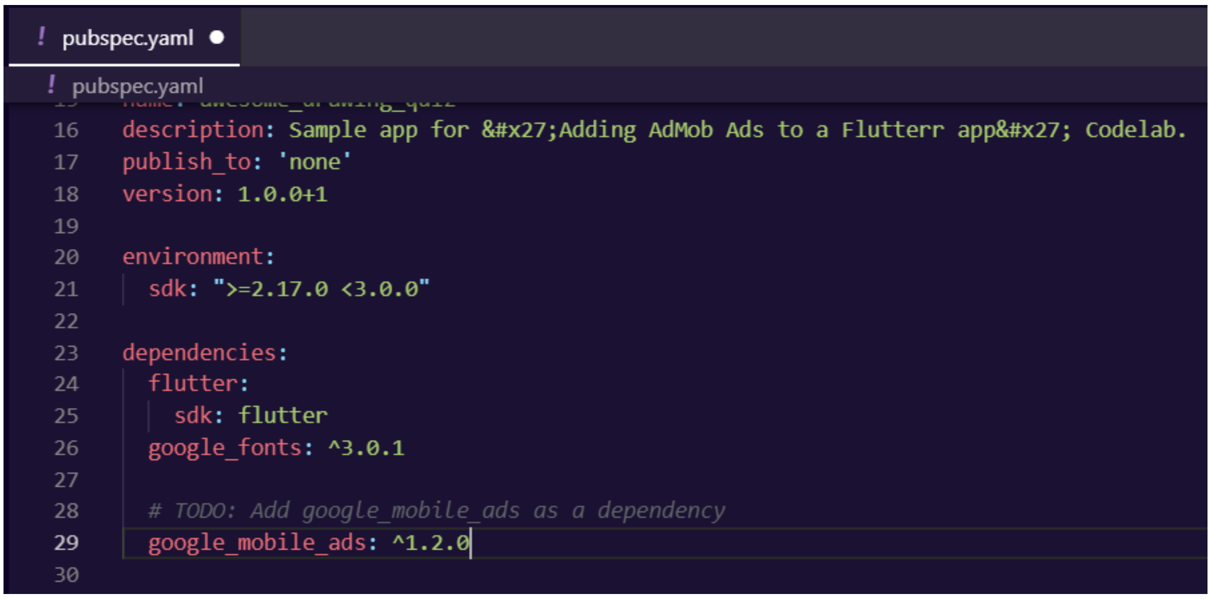
Add the Google Mobile Ads Flutter plugin

Flutter uses plugins to provide access to a wide range of platform-specific services. Plugins enable us to access services and APIs on each platform.

The google_mobile_ads plugin supports loading and displaying banner, interstitial, rewarded, and native ads using the AdMob API.

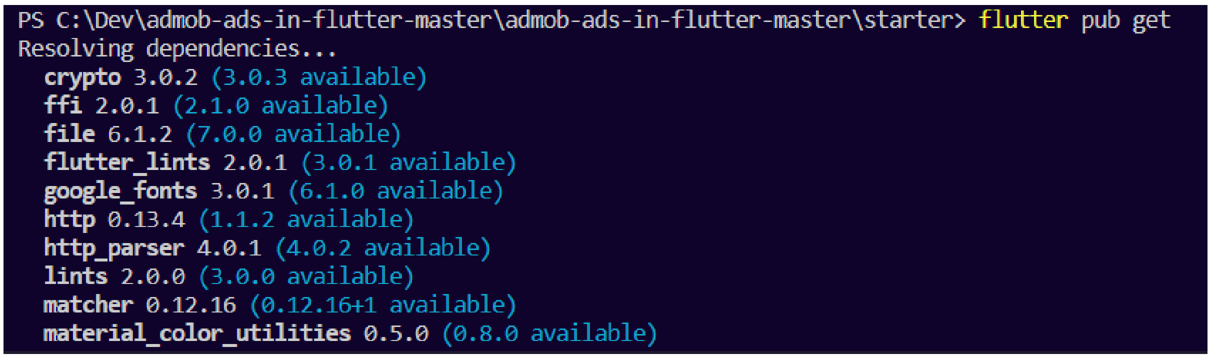
Add the Google Mobile Ads plugin as a dependency

Step 1: To access the AdMob APIs from the AdMob inline ads project, add google_mobile_ads as a dependency to the pubspec.yaml file located at the root of the project.



```yaml
description: Sample app for &#x27;Adding AdMob Ads to a Flutterr app&#x27; Codelab.
publish_to: 'none'
version: 1.0.0+1

environment:
  sdk: ">=2.17.0 <3.0.0"

dependencies:
  flutter:
    sdk: flutter
  google_fonts: ^3.0.1

  # TODO: Add google_mobile_ads as a dependency
  google_mobile_ads: ^1.2.0
```

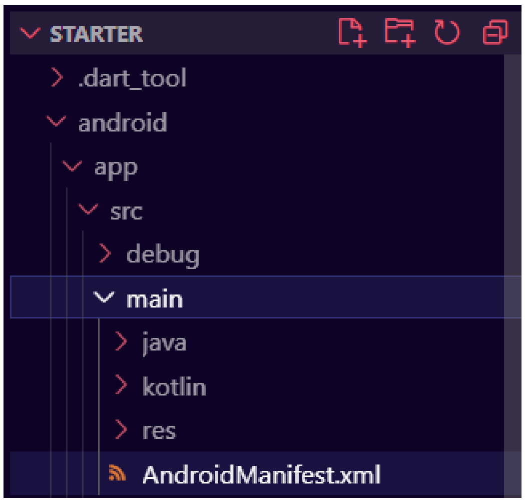
Step 2: Click Pub get to install the plugin in the Awesome Drawing Quiz project.



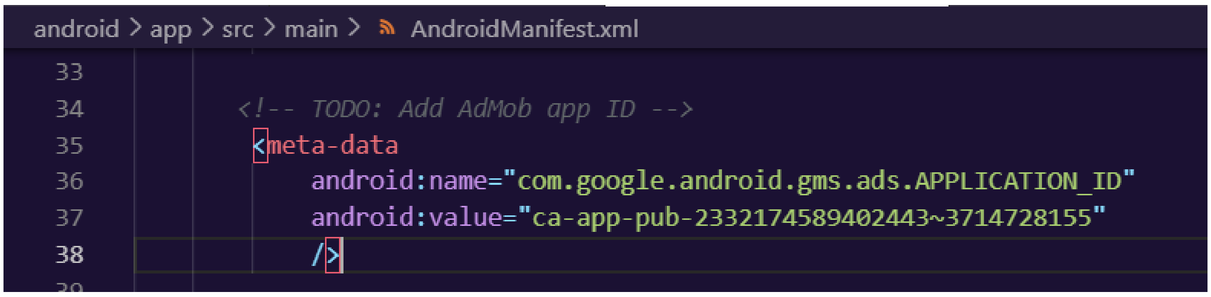
```
PS C:\Dev\admob-ads-in-flutter-master\admob-ads-in-flutter-master\starter> flutter pub get
Resolving dependencies...
  crypto 3.0.2 (3.0.3 available)
  ffi 2.0.1 (2.1.0 available)
  file 6.1.2 (7.0.0 available)
  flutter_lints 2.0.1 (3.0.1 available)
  google_fonts 3.0.1 (6.1.0 available)
  http 0.13.4 (1.1.2 available)
  http_parser 4.0.1 (4.0.2 available)
  lints 2.0.0 (3.0.0 available)
  matcher 0.12.16 (0.12.16+1 available)
  material_color_utilities 0.5.0 (0.8.0 available)
```

Update AndroidManifest.xml (Android)

Step 1: Open the android/app/src/main/AndroidManifest.xml file in Android Studio.



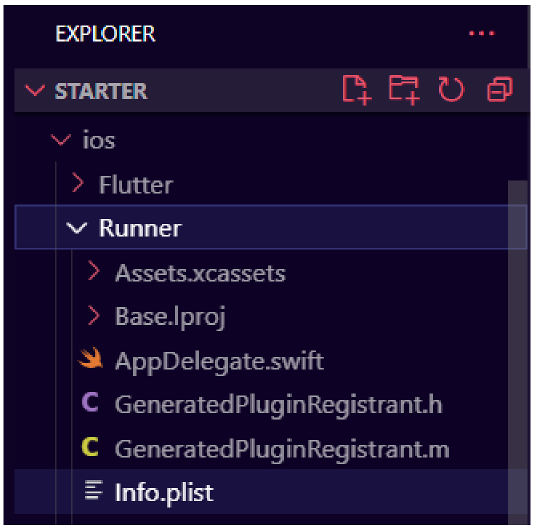
Step 2: Add the AdMob app ID by adding a <meta-data> tag with the name com.google.android.gms.ads.APPLICATION_ID**.**


android > app > src > main > AndroidManifest.xml

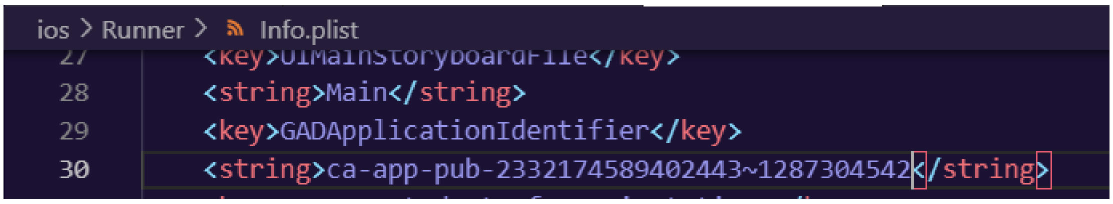
```
    <!-- TODO: Add AdMob app ID -->
    <meta-data
        android:name="com.google.android.gms.ads.APPLICATION_ID"
        android:value="ca-app-pub-2332174589402443~3714728155"
        />
```

Update Info.plist (iOS)

Step 1: Open the ios/Runner/Info.plist file in Android Studio.



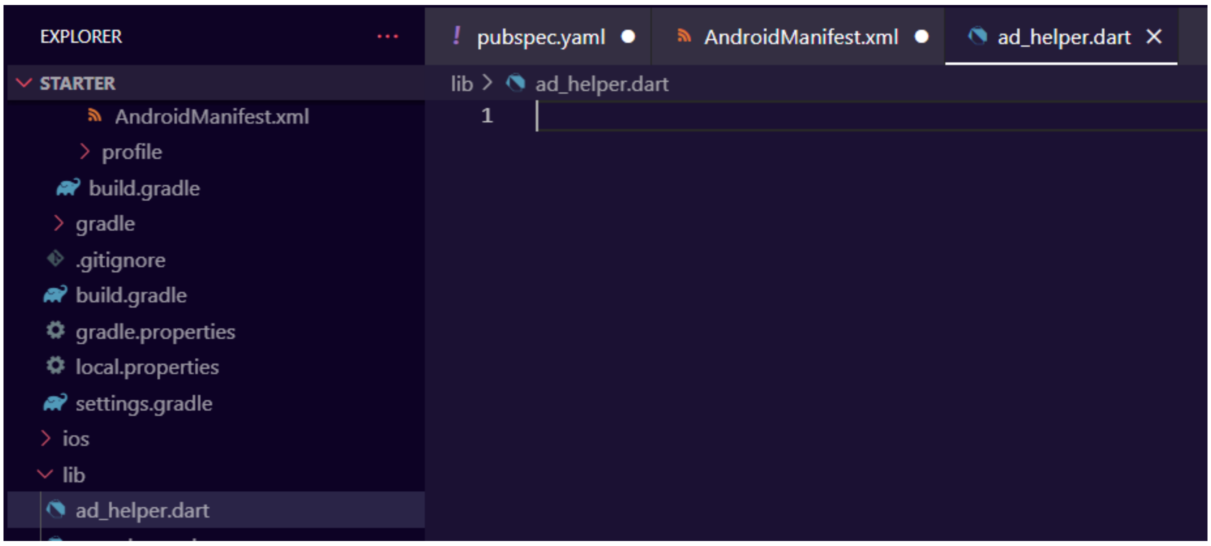
Step 2: Add a GADApplicationIdentifier key with the string value of your AdMob app ID.


ios > Runner > Info.plist

```
<key>UIMainStoryboardFile</key>
<string>Main</string>
<key>GADApplicationIdentifier</key>
<string>ca-app-pub-2332174589402443~1287304542</string>
```

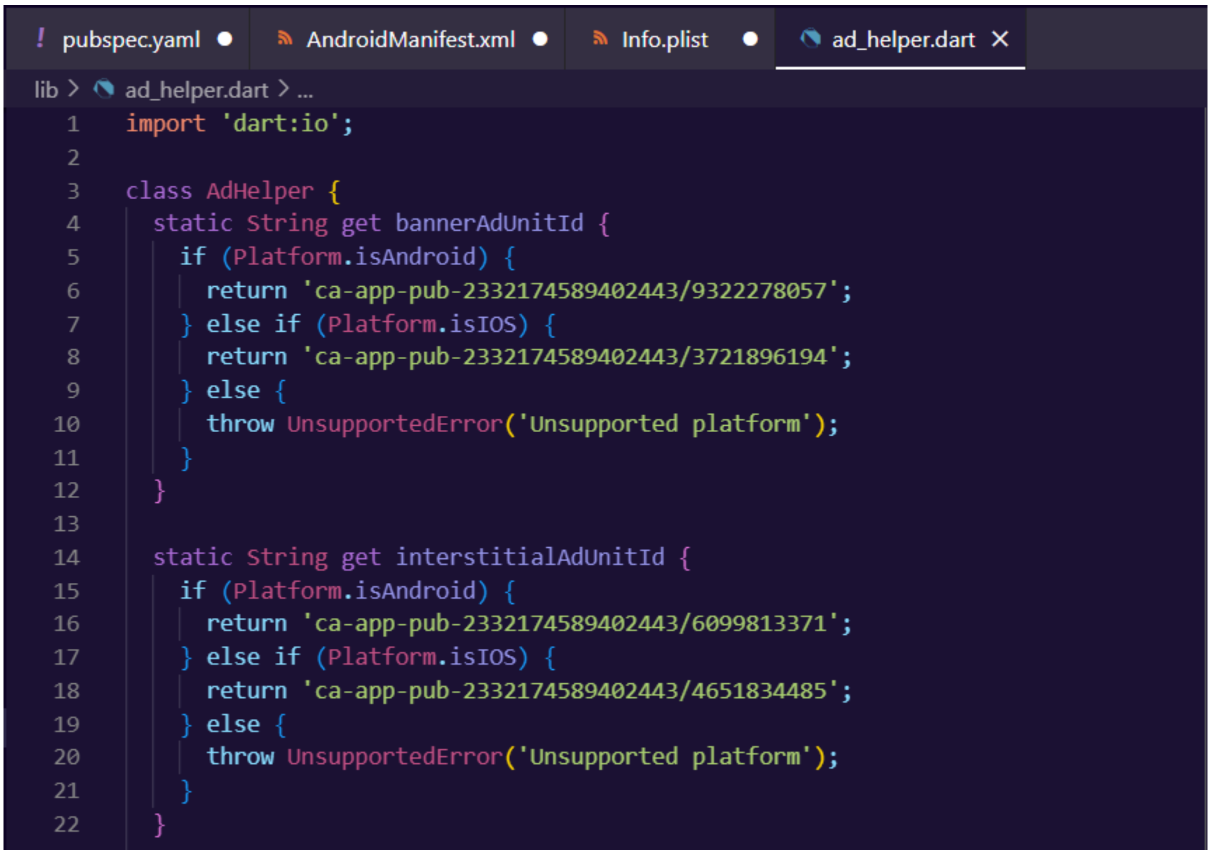
Add a helper class for ads

Step 1: Create a new file named ad_helper.dart under the lib directory.



Step 2: Implement the AdHelper class, which provides an AdMob app ID and ad unit IDs for Android.

Make sure to replace the AdMob app ID (ca-app-pub-xxxxxx~yyyyy) and the ad unit ID (ca-app-pub-xxxxxxx/yyyyyyyy) with the IDs created in the previous step.

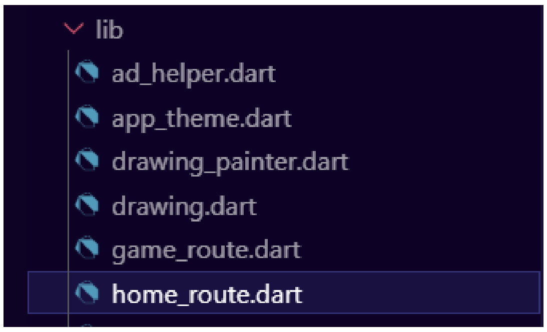


```dart
import 'dart:io';

class AdHelper {
  static String get bannerAdUnitId {
    if (Platform.isAndroid) {
      return 'ca-app-pub-2332174589402443/9322278057';
    } else if (Platform.isIOS) {
      return 'ca-app-pub-2332174589402443/3721896194';
    } else {
      throw UnsupportedError('Unsupported platform');
    }
  }

  static String get interstitialAdUnitId {
    if (Platform.isAndroid) {
      return 'ca-app-pub-2332174589402443/6099813371';
    } else if (Platform.isIOS) {
      return 'ca-app-pub-2332174589402443/4651834485';
    } else {
      throw UnsupportedError('Unsupported platform');
    }
  }
```

Initialize the Google Mobile Ads SDK

Before loading ads, you need to initialize the Google Mobile Ads SDK.

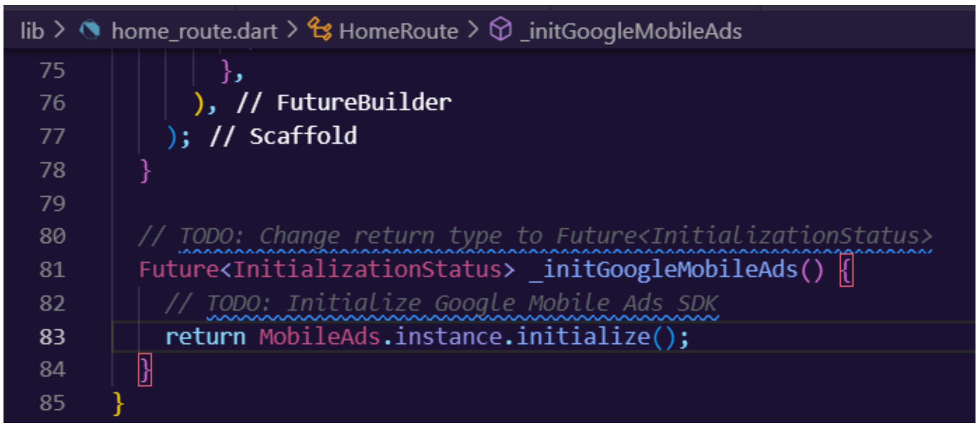
Step 1: Open the lib/home_route.dart file



```
lib
  ad_helper.dart
  app_theme.dart
  drawing_painter.dart
  drawing.dart
  game_route.dart
  home_route.dart
```

Step 2: Modify _initGoogleMobileAds() to initialize the SDK before the home page is loaded.

Note: We need to change the return type of the _initGoogleMobileAds() method from Future<dynamic> to Future<InitializationStatus> to get the SDK initialization result after it completes.

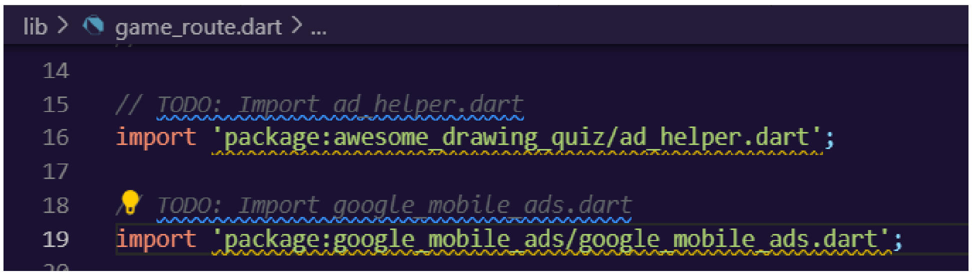


```
lib > home_route.dart > HomeRoute > _initGoogleMobileAds
          },
        ), // FutureBuilder
      ); // Scaffold
    }

    // TODO: Change return type to Future<InitializationStatus>
    Future<InitializationStatus> _initGoogleMobileAds() {
      // TODO: Initialize Google Mobile Ads SDK
      return MobileAds.instance.initialize();
    }
  }
```

Add a banner ad

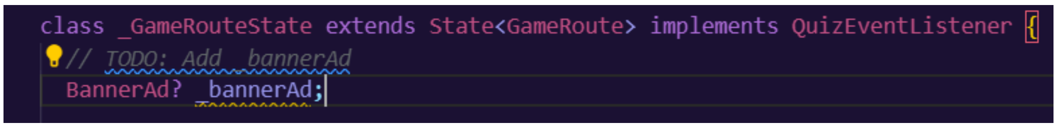
Step 1: Open the lib/game_route.dart file, and import ad_manager.dart

Step 2: Import the ad_helper.dart and google_mobile_ads.dart by adding the following lines:



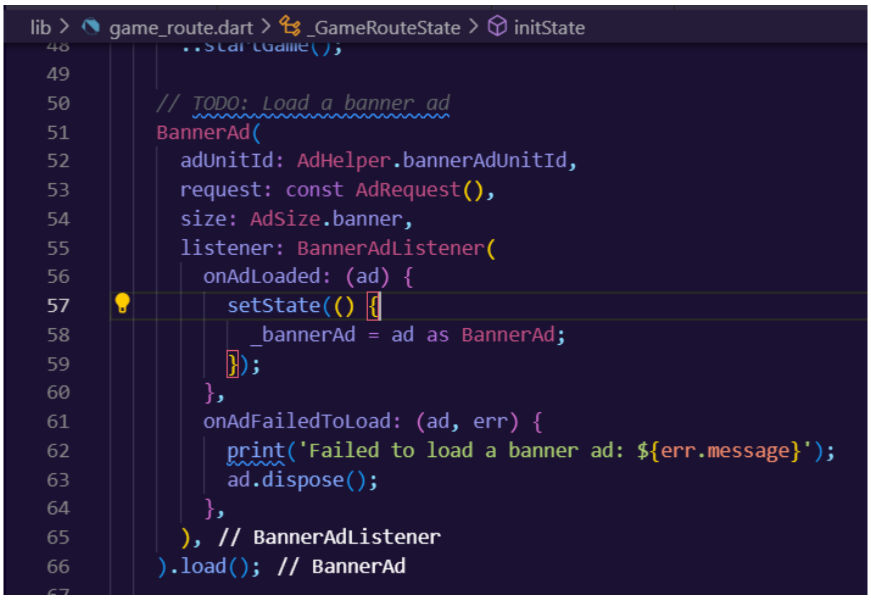
```
lib > game_route.dart > ...

// TODO: Import ad_helper.dart
import 'package:awesome_drawing_quiz/ad_helper.dart';

// TODO: Import google_mobile_ads.dart
import 'package:google_mobile_ads/google_mobile_ads.dart';
```

Step 3: In the _GameRouteState class, add the following members for a banner ad.



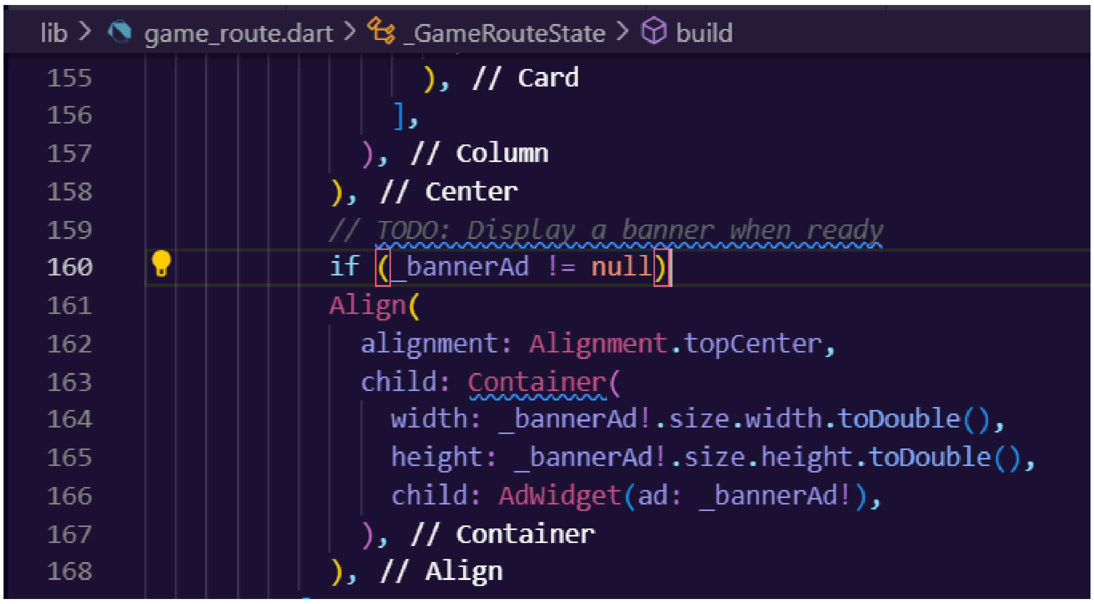
```
class _GameRouteState extends State<GameRoute> implements QuizEventListener {
  // TODO: Add _bannerAd
  BannerAd? _bannerAd;
```

Step 4: In the initState() method, create and load a BannerAd for the 320x50 banner (AdSize.banner). Note that an ad event listener is configured to update the UI (setState()) when an ad is loaded .


lib > game_route.dart > _GameRouteState > initState

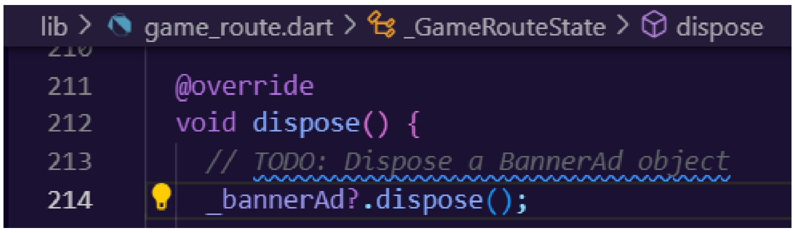
```
      ..startGame();

      // TODO: Load a banner ad
      BannerAd(
        adUnitId: AdHelper.bannerAdUnitId,
        request: const AdRequest(),
        size: AdSize.banner,
        listener: BannerAdListener(
          onAdLoaded: (ad) {
            setState(() {
              _bannerAd = ad as BannerAd;
            });
          },
          onAdFailedToLoad: (ad, err) {
            print('Failed to load a banner ad: ${err.message}');
            ad.dispose();
          },
        ), // BannerAdListener
      ).load(); // BannerAd
```

Step 5: Modify the build() method to display a banner ad when available.


lib > game_route.dart > _GameRouteState > build

```
                  ), // Card
                ],
              ), // Column
            ), // Center
            // TODO: Display a banner when ready
            if (_bannerAd != null)
            Align(
              alignment: Alignment.topCenter,
              child: Container(
                width: _bannerAd!.size.width.toDouble(),
                height: _bannerAd!.size.height.toDouble(),
                child: AdWidget(ad: _bannerAd!),
              ), // Container
            ), // Align
```

Step 6: Release the resource associated with the BannerAd object by calling the BannerAd.dispose() method in the dispose() callback method.


lib > game_route.dart > _GameRouteState > dispose

```
  @override
  void dispose() {
    // TODO: Dispose a BannerAd object
    _bannerAd?.dispose();
```

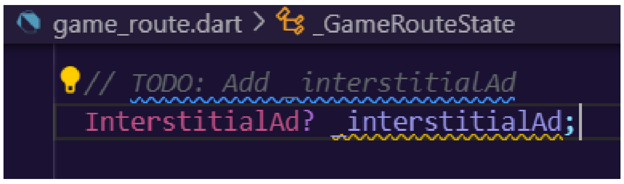
Step 7: Run the project, and start a new game. After an ad is loaded, you'll see a banner ad at the top of the screen.

Add an interstitial ad

In this section, we display an interstitial ad after the game (5 levels in total) finishes.

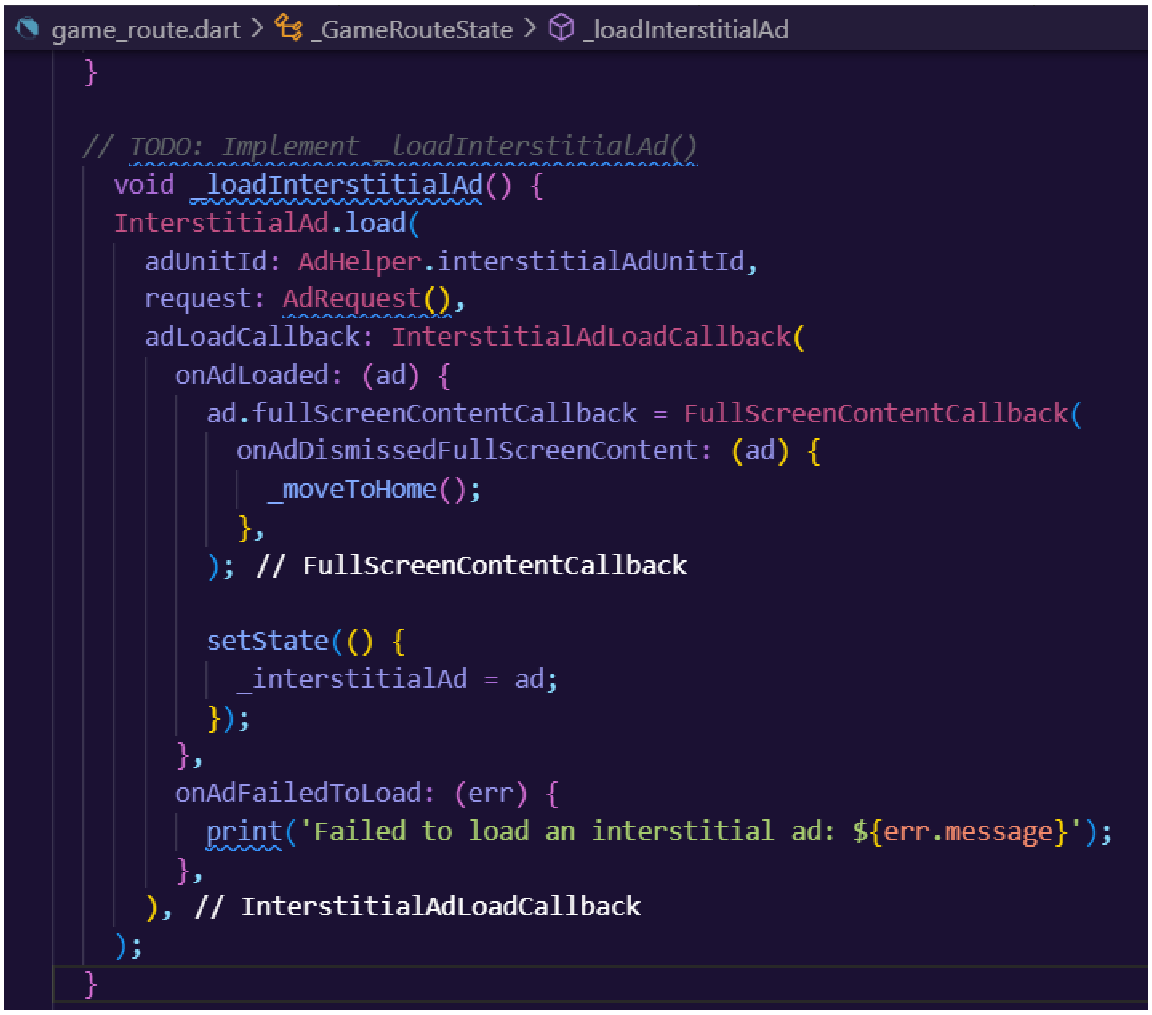
Step 1: Open the lib/game_route.dart file

Step 2: In the _GameRouteState class, add the following members and methods for an interstitial ad.



game_route.dart > _GameRouteState

```
// TODO: Add _interstitialAd
InterstitialAd? _interstitialAd;
```



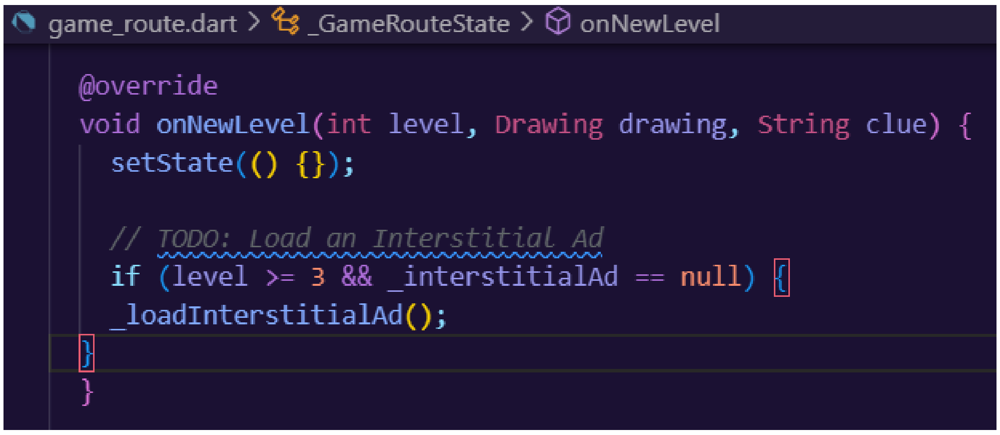
game_route.dart > _GameRouteState > _loadInterstitialAd

```
}

// TODO: Implement _loadInterstitialAd()
  void _loadInterstitialAd() {
  InterstitialAd.load(
    adUnitId: AdHelper.interstitialAdUnitId,
    request: AdRequest(),
    adLoadCallback: InterstitialAdLoadCallback(
      onAdLoaded: (ad) {
        ad.fullScreenContentCallback = FullScreenContentCallback(
          onAdDismissedFullScreenContent: (ad) {
            _moveToHome();
          },
        ); // FullScreenContentCallback

        setState(() {
          _interstitialAd = ad;
        });
      },
      onAdFailedToLoad: (err) {
        print('Failed to load an interstitial ad: ${err.message}');
      },
    ), // InterstitialAdLoadCallback
  );
}
```

Note that an ad event listener is configured to check whether the ad is ready (onAdLoaded() and onAdFailedToLoad()) and to display the app's home screen when the ad is closed (onAdDismissedFullScreenContent()).

Step 3: In this codelab, an interstitial ad is displayed after a user completes 5 levels. To minimize unnecessary ad requests, request an ad when a user reaches level 3.

In the onNewLevel() method, add the following lines.

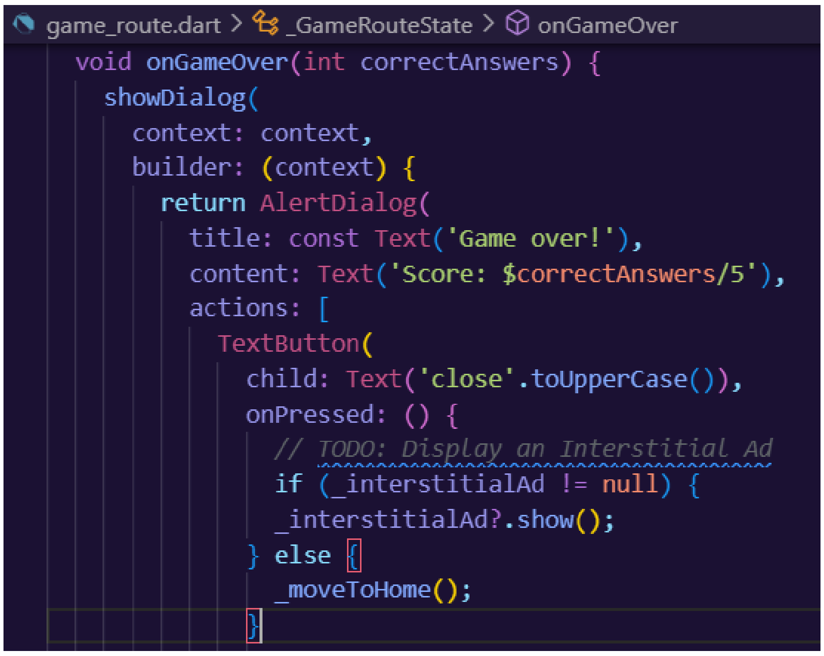


```dart
@override
void onNewLevel(int level, Drawing drawing, String clue) {
  setState(() {});

  // TODO: Load an Interstitial Ad
  if (level >= 3 && _interstitialAd == null) {
  _loadInterstitialAd();
}
}
```

Step 4: When a game finishes, the game score dialog is displayed. When a user closes the dialog, it routes a user to the home screen of the Awesome Drawing Quiz.

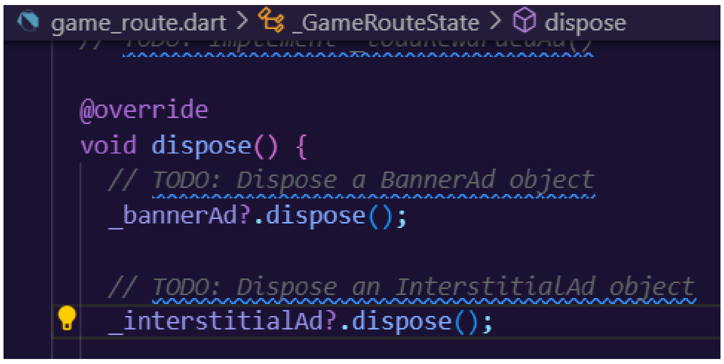
Because interstitial ads should be displayed between screen transitions, we show the interstitial ad when a user clicks the CLOSE button.

Modify the onGameOver() method as follows.



```dart
void onGameOver(int correctAnswers) {
  showDialog(
    context: context,
    builder: (context) {
      return AlertDialog(
        title: const Text('Game over!'),
        content: Text('Score: $correctAnswers/5'),
        actions: [
          TextButton(
            child: Text('close'.toUpperCase()),
            onPressed: () {
              // TODO: Display an Interstitial Ad
              if (_interstitialAd != null) {
              _interstitialAd?.show();
            } else {
              _moveToHome();
            }
```

Step 5: Release the resource associated with the InterstitialAd object by calling the InterstitialAd.dispose() method in the dispose() callback method.



```dart
@override
void dispose() {
  // TODO: Dispose a BannerAd object
  _bannerAd?.dispose();

  // TODO: Dispose an InterstitialAd object
  _interstitialAd?.dispose();
```

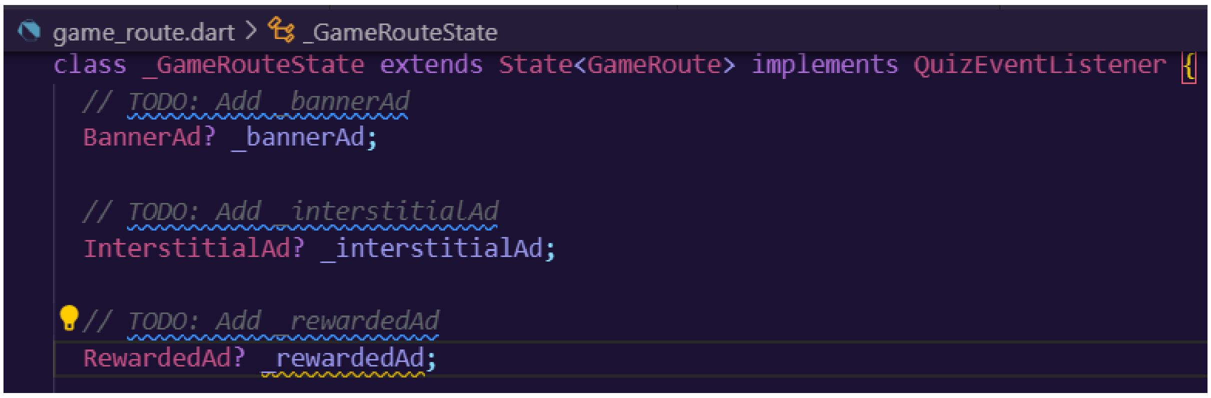
Step 6: Run the project and complete the game. If an interstitial ad is loaded, you'll see an interstitial ad once you click CLOSE button from the score dialog.

Add a rewarded ad

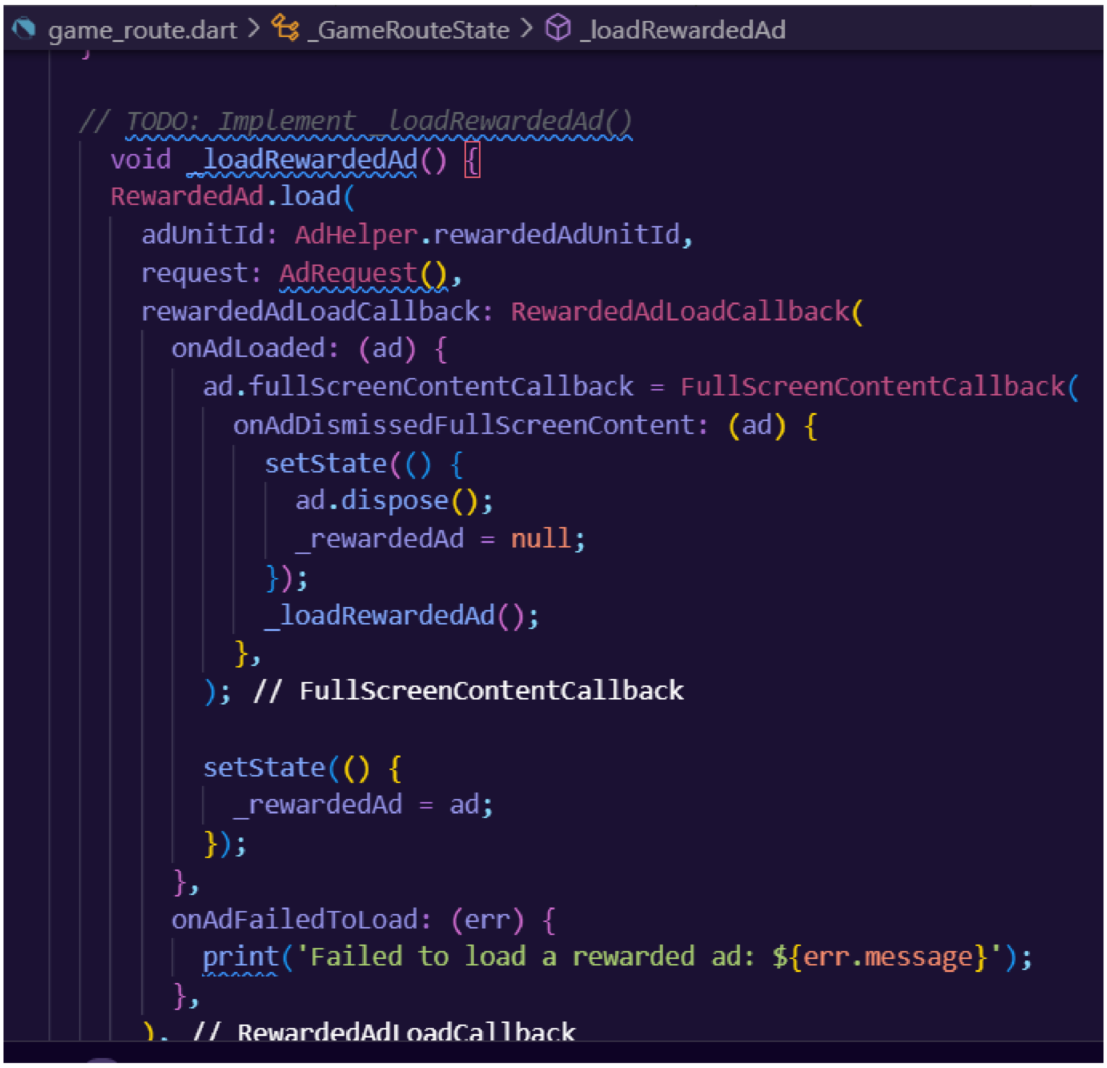
In this section, we add a rewarded ad which gives a user an additional hint as a reward.

Step 1: Open the lib/game_route.dart file

Step 2: In the _GameRouteState class, add members for a rewarded ad, and implement _loadRewardedAd() method.



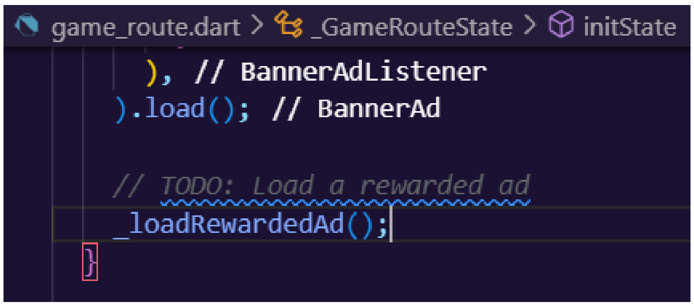
```
game_route.dart > _GameRouteState
class _GameRouteState extends State<GameRoute> implements QuizEventListener {
  // TODO: Add _bannerAd
  BannerAd? _bannerAd;

  // TODO: Add _interstitialAd
  InterstitialAd? _interstitialAd;

  // TODO: Add _rewardedAd
  RewardedAd? _rewardedAd;
```



```
game_route.dart > _GameRouteState > _loadRewardedAd

  // TODO: Implement _loadRewardedAd()
   void _loadRewardedAd() {
   RewardedAd.load(
     adUnitId: AdHelper.rewardedAdUnitId,
     request: AdRequest(),
     rewardedAdLoadCallback: RewardedAdLoadCallback(
       onAdLoaded: (ad) {
         ad.fullScreenContentCallback = FullScreenContentCallback(
           onAdDismissedFullScreenContent: (ad) {
             setState(() {
               ad.dispose();
               _rewardedAd = null;
             });
             _loadRewardedAd();
           },
         ); // FullScreenContentCallback

         setState(() {
           _rewardedAd = ad;
         });
       },
       onAdFailedToLoad: (err) {
         print('Failed to load a rewarded ad: ${err.message}');
       },
     ), // RewardedAdLoadCallback
```

Note that it loads another rewarded ad when the ad is closed (onAdDismissedFullScreenContent) to cache the ad as early as possible.

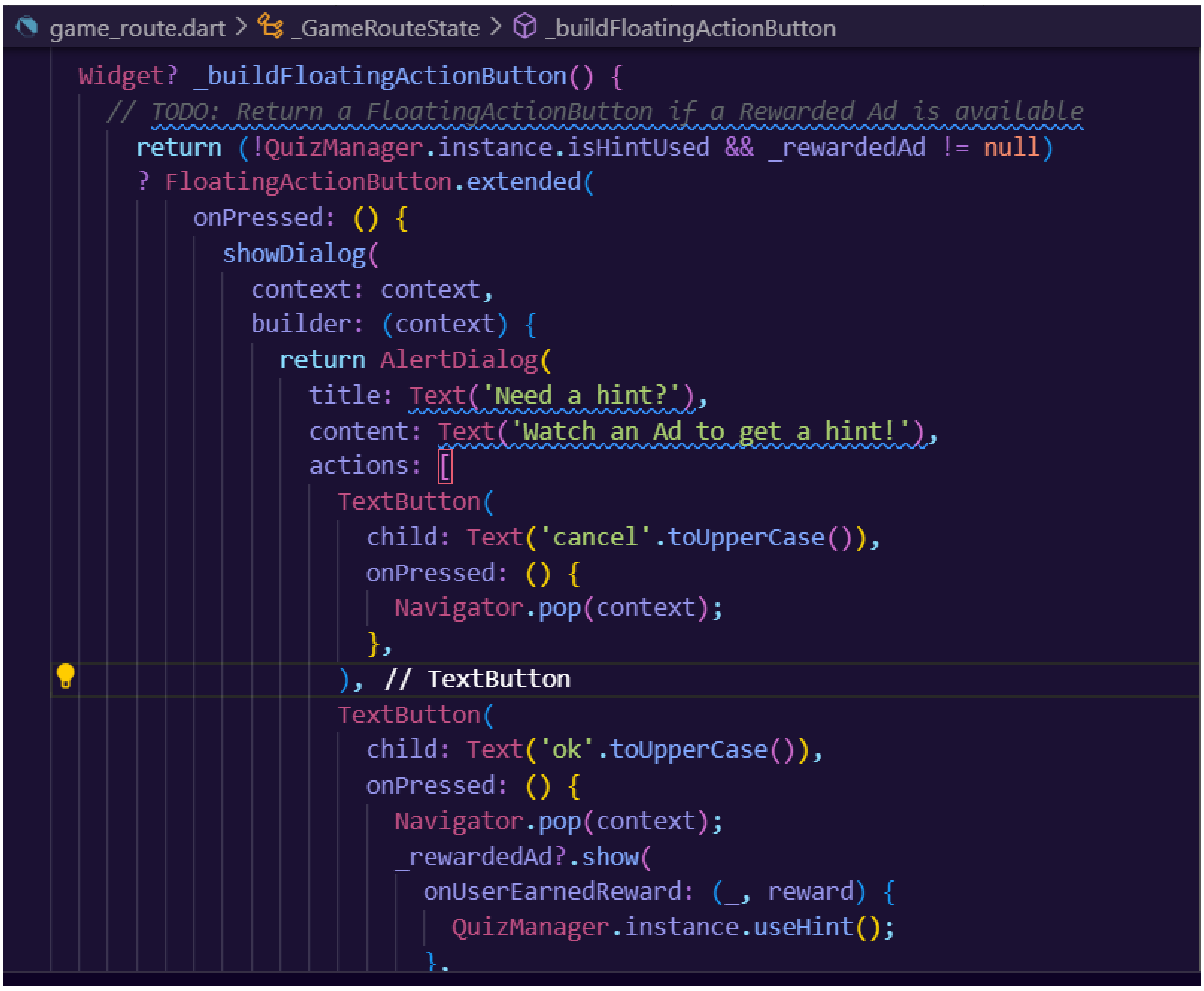
Step 3: Call _loadRewardedAd() from initState() method to request a rewarded ad when the game starts.



```
game_route.dart > _GameRouteState > initState
      ), // BannerAdListener
    ).load(); // BannerAd

    // TODO: Load a rewarded ad
    _loadRewardedAd();
  }
```

Step 4: Allow users to watch a rewarded ad by clicking the floating action button. The button shows only if a user hasn't used a hint at the current level and a rewarded ad is loaded.

Modify the _buildFloatingActionButton() method, as follows, to display the floating action button. Note that returning null hides the button from the screen.

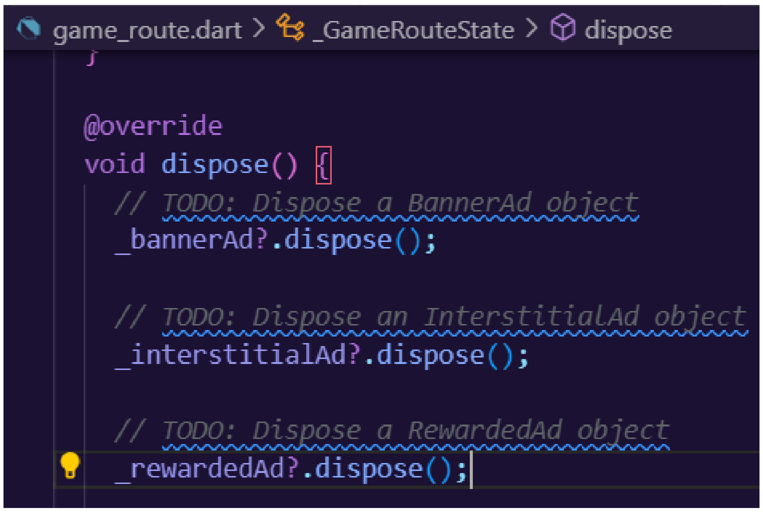


```
game_route.dart > _GameRouteState > _buildFloatingActionButton
  Widget? _buildFloatingActionButton() {
    // TODO: Return a FloatingActionButton if a Rewarded Ad is available
    return (!QuizManager.instance.isHintUsed && _rewardedAd != null)
      ? FloatingActionButton.extended(
          onPressed: () {
            showDialog(
              context: context,
              builder: (context) {
                return AlertDialog(
                  title: Text('Need a hint?'),
                  content: Text('Watch an Ad to get a hint!'),
                  actions: [
                    TextButton(
                      child: Text('cancel'.toUpperCase()),
                      onPressed: () {
                        Navigator.pop(context);
                      },
                    ), // TextButton
                    TextButton(
                      child: Text('ok'.toUpperCase()),
                      onPressed: () {
                        Navigator.pop(context);
                        _rewardedAd?.show(
                          onUserEarnedReward: (_, reward) {
                            QuizManager.instance.useHint();
                          },
```

Note that onUserEarnedReward is the most important ad event in a rewarded ad. It's triggered when a user becomes eligible to receive a reward (for example., finished watching a video).

In this codelab, the QuizManager.instance.useHint() method is called from the callback, which reveals one more character in the hint string. The app reloads a rewarded ad in the onAdClosed callback to make sure the ad is ready as early as possible.

Step 5: Release the resource associated with the RewardedAd object by calling the RewardedAd.dispose() method in the dispose() callback method.



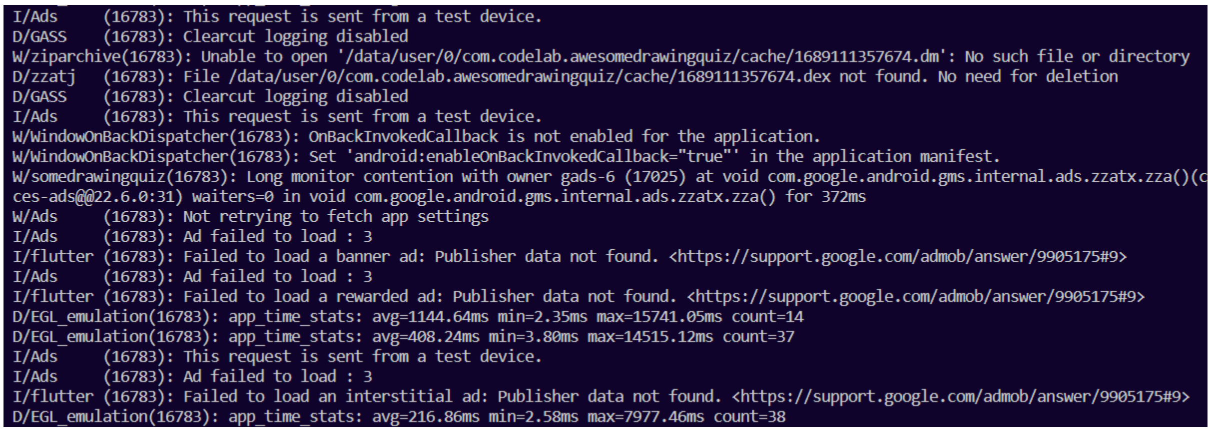
```
game_route.dart > _GameRouteState > dispose

  @override
  void dispose() {
    // TODO: Dispose a BannerAd object
    _bannerAd?.dispose();

    // TODO: Dispose an InterstitialAd object
    _interstitialAd?.dispose();

    // TODO: Dispose a RewardedAd object
    _rewardedAd?.dispose();
```

Step 6: Run the project and play the game. Once a rewarded ad is loaded, you'll see a hint button at the bottom of the screen. Click Hint button to get an additional hint.

Testing Ads:

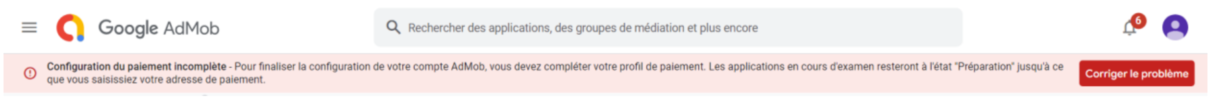
The application runs successfully but ads are not loaded.

We can see these messages in the terminal:



```
I/Ads     (16783): This request is sent from a test device.
D/GASS    (16783): Clearcut logging disabled
W/ziparchive(16783): Unable to open '/data/user/0/com.codelab.awesomedrawingquiz/cache/1689111357674.dm': No such file or directory
D/zzatj   (16783): File /data/user/0/com.codelab.awesomedrawingquiz/cache/1689111357674.dex not found. No need for deletion
D/GASS    (16783): Clearcut logging disabled
I/Ads     (16783): This request is sent from a test device.
W/WindowOnBackDispatcher(16783): OnBackInvokedCallback is not enabled for the application.
W/WindowOnBackDispatcher(16783): Set 'android:enableOnBackInvokedCallback="true"' in the application manifest.
W/somedrawingquiz(16783): Long monitor contention with owner gads-6 (17025) at void com.google.android.gms.internal.ads.zzatx.zza()(c
ces-ads@@22.6.0:31) waiters=0 in void com.google.android.gms.internal.ads.zzatx.zza() for 372ms
W/Ads     (16783): Not retrying to fetch app settings
I/Ads     (16783): Ad failed to load : 3
I/flutter (16783): Failed to load a banner ad: Publisher data not found. <https://support.google.com/admob/answer/9905175#9>
I/Ads     (16783): Ad failed to load : 3
I/flutter (16783): Failed to load a rewarded ad: Publisher data not found. <https://support.google.com/admob/answer/9905175#9>
D/EGL_emulation(16783): app_time_stats: avg=1144.64ms min=2.35ms max=15741.05ms count=14
D/EGL_emulation(16783): app_time_stats: avg=408.24ms min=3.80ms max=14515.12ms count=37
I/Ads     (16783): This request is sent from a test device.
I/Ads     (16783): Ad failed to load : 3
I/flutter (16783): Failed to load an interstitial ad: Publisher data not found. <https://support.google.com/admob/answer/9905175#9>
D/EGL_emulation(16783): app_time_stats: avg=216.86ms min=2.58ms max=7977.46ms count=38
```

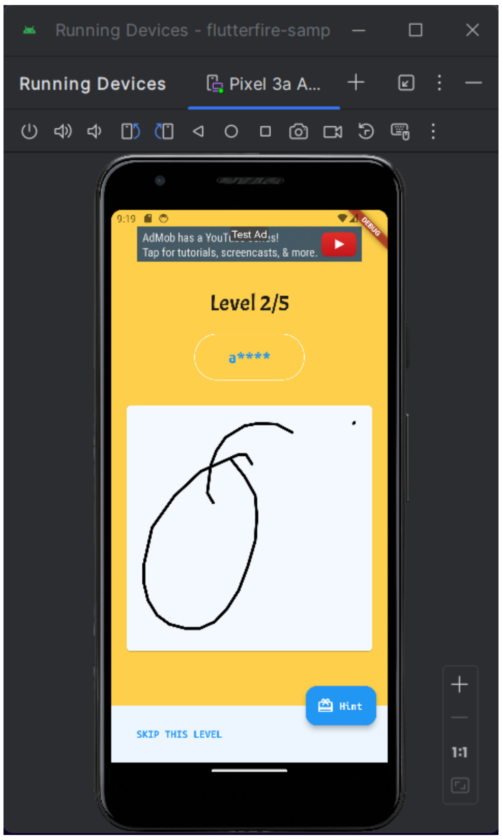
The error message "Publisher data not found" indicates that the publisher data for the ad unit cannot be found. There might be an issue with our Google AdMob account as we did not fill the payment section :


Google AdMob
Rechercher des applications, des groupes de médiation et plus encore
6
Corriger le problème

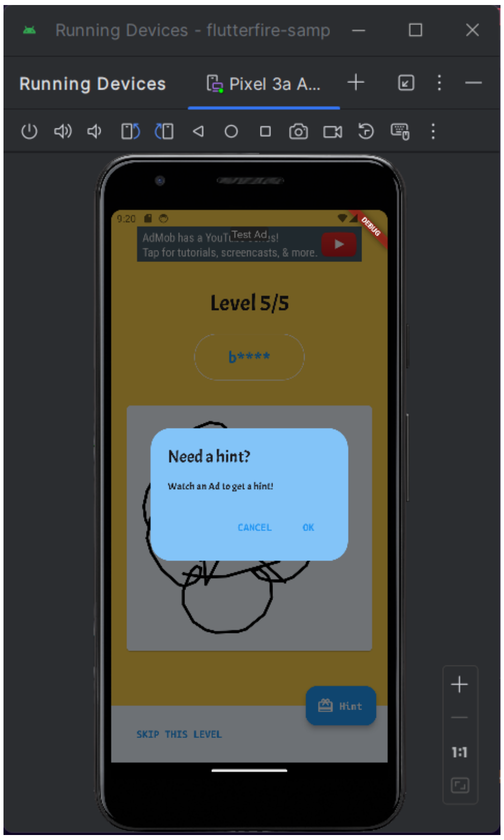
Configuration du paiement incomplète - Pour finaliser la configuration de votre compte AdMob, vous devez compléter votre profil de paiement. Les applications en cours d'examen resteront à l'état "Préparation" jusqu'à ce que vous saisissiez votre adresse de paiement.

As we do not want to complete the payment configuration, we will use the testing IDs provided by the codelab.

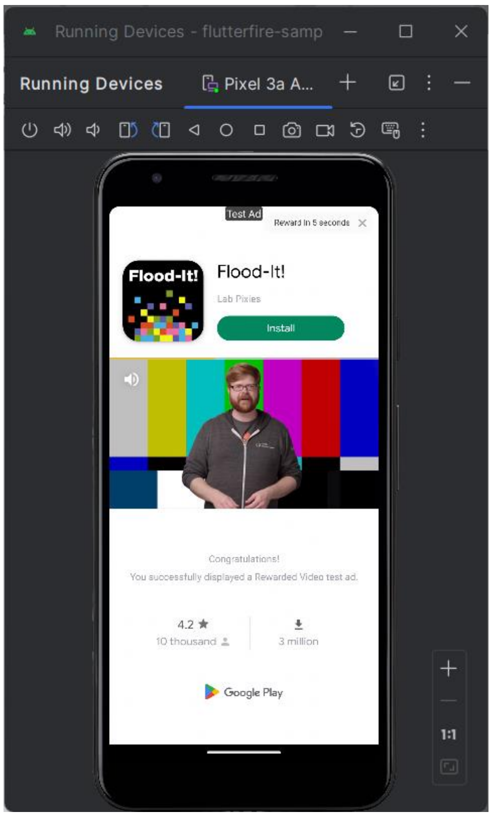
As we can see below, the banner ad appears on the top of the application.


Running Devices - flutterfire-samp
Running Devices
Pixel 3a A...
Test Ad
AdMob has a YouTube channel!
Tap for tutorials, screencasts, & more.
Level 2/5
a****
Hint
SKIP THIS LEVEL
1:1

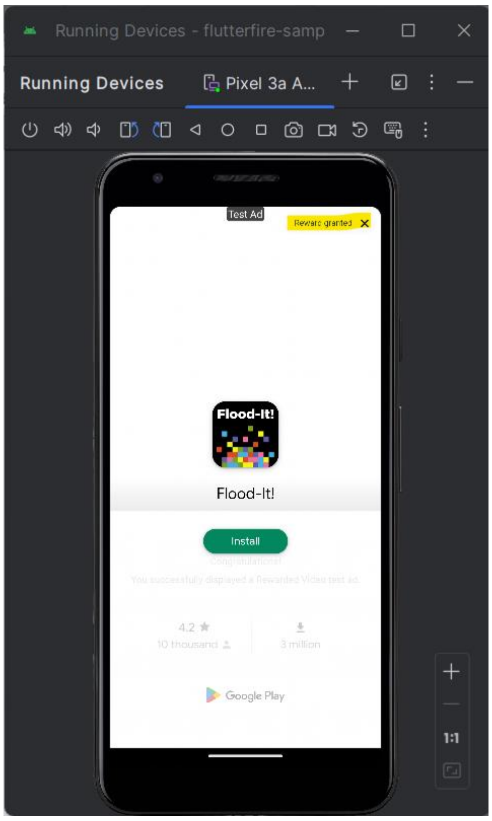
Next, upon clicking on the 'Hint' button, we can see the following pop-up window:



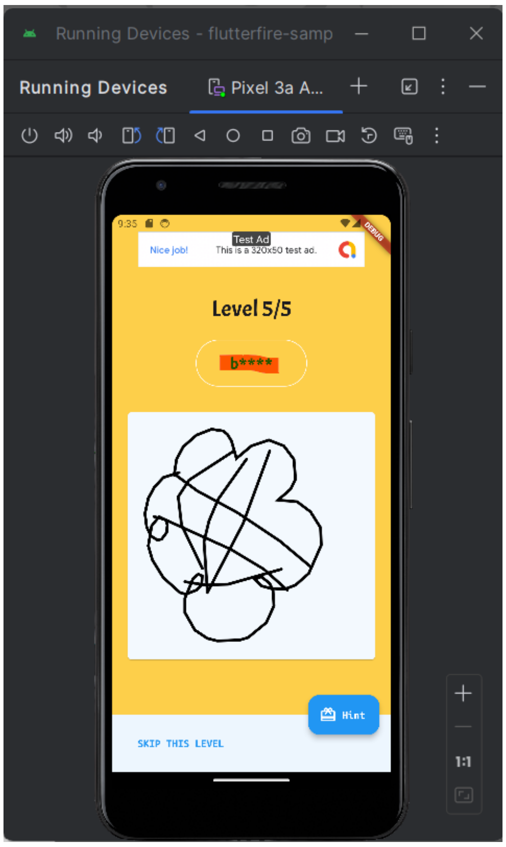
If we click 'OK', the rewarded ad is launched:



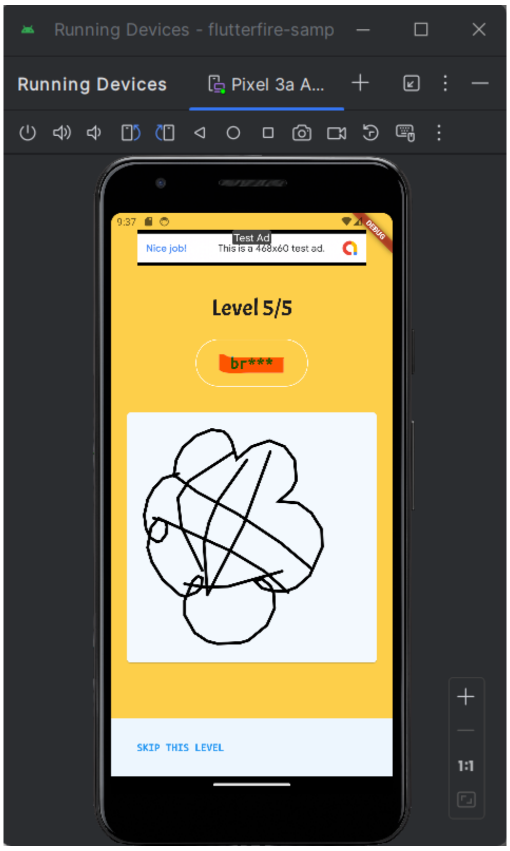
After a defined time laps, a message 'Reward granted' appears on the top right next to the close button:



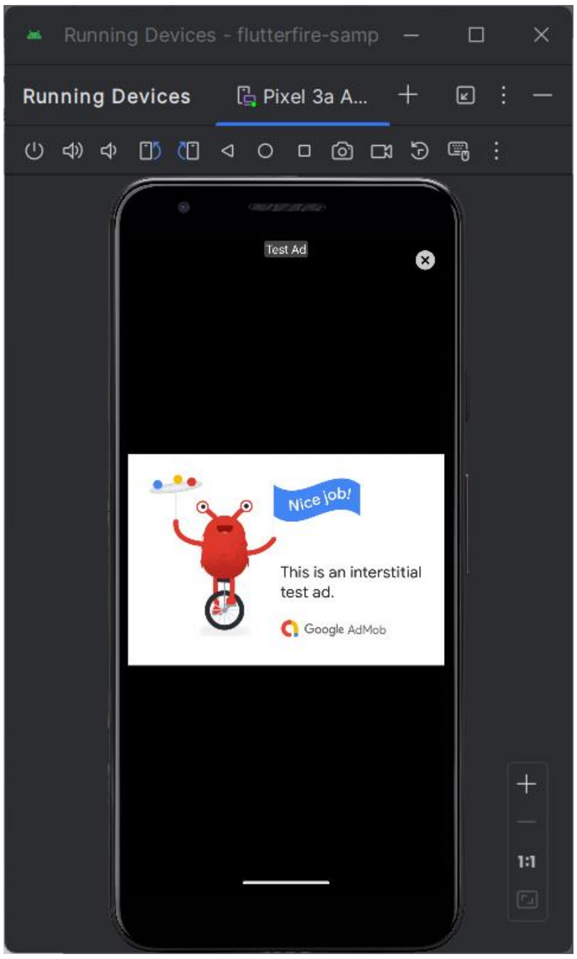
Before clicking on the 'Hint' button, highlighted in red below was the text displayed:



After the rewarded ad has been granted, we can see that a letter as been added :



When game is over, we click on the close button and the interstitial ad is loaded:


Running Devices - flutterfire-samp
Running Devices
Pixel 3a A...
Test Ad
Nice job!
This is an interstitial test ad.
Google AdMob
1:1

Conclusion

In conclusion, this tutorial provided a comprehensive overview of adding ads in mobile apps using Google AdMob. We covered the process of creating a Google AdMob account, setting up ad units, and implementing them in our mobile app. We discussed the various types of ads available and their constraints.

Additionally, we explored common errors that may occur during the implementation of ads and provided guidance on how to debug and resolve these issues. From account approval delays to ad unit configuration mismatches, we addressed key onboarding issues and their solutions.

Implementing ads in mobile apps can be a complex task, but with the knowledge gained from this tutorial, we are well-equipped to successfully navigate the process and maximize their app's revenue potential.

References

https://codelabs.developers.google.com/codelabs/admob-ads-in-flutter#9

https://apps.admob.com/v2/apps/3714728155/adunits/list