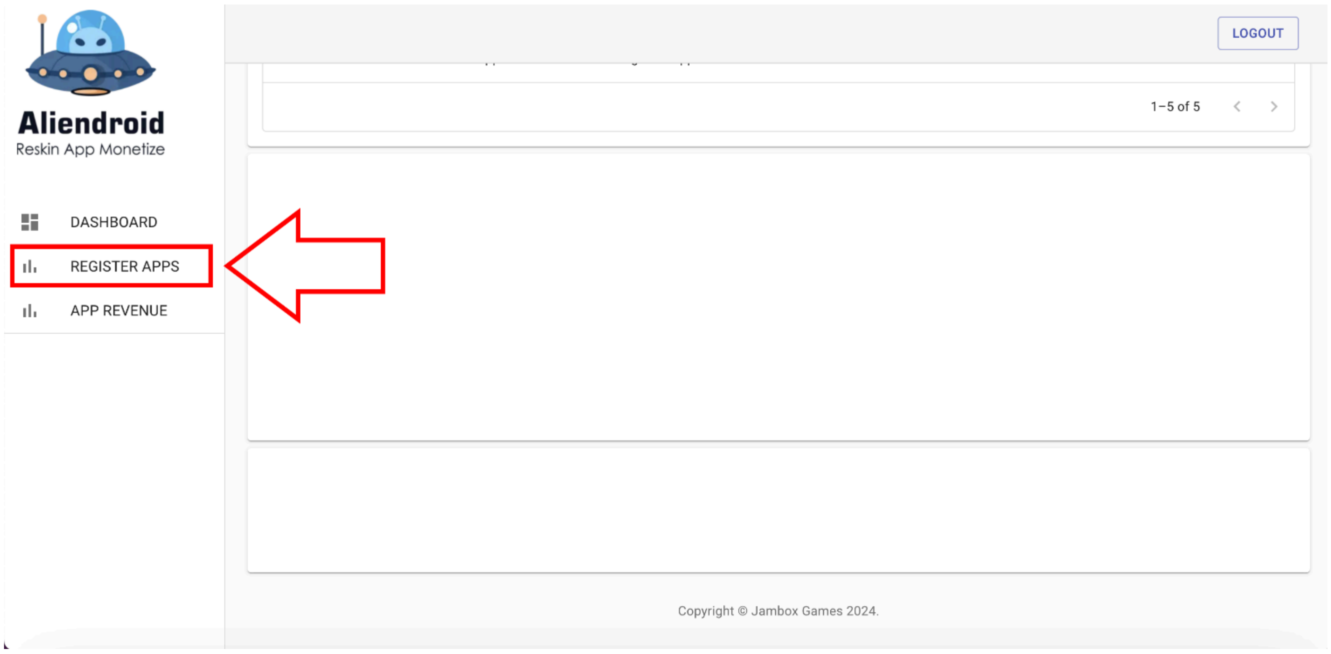
Jambox SDK

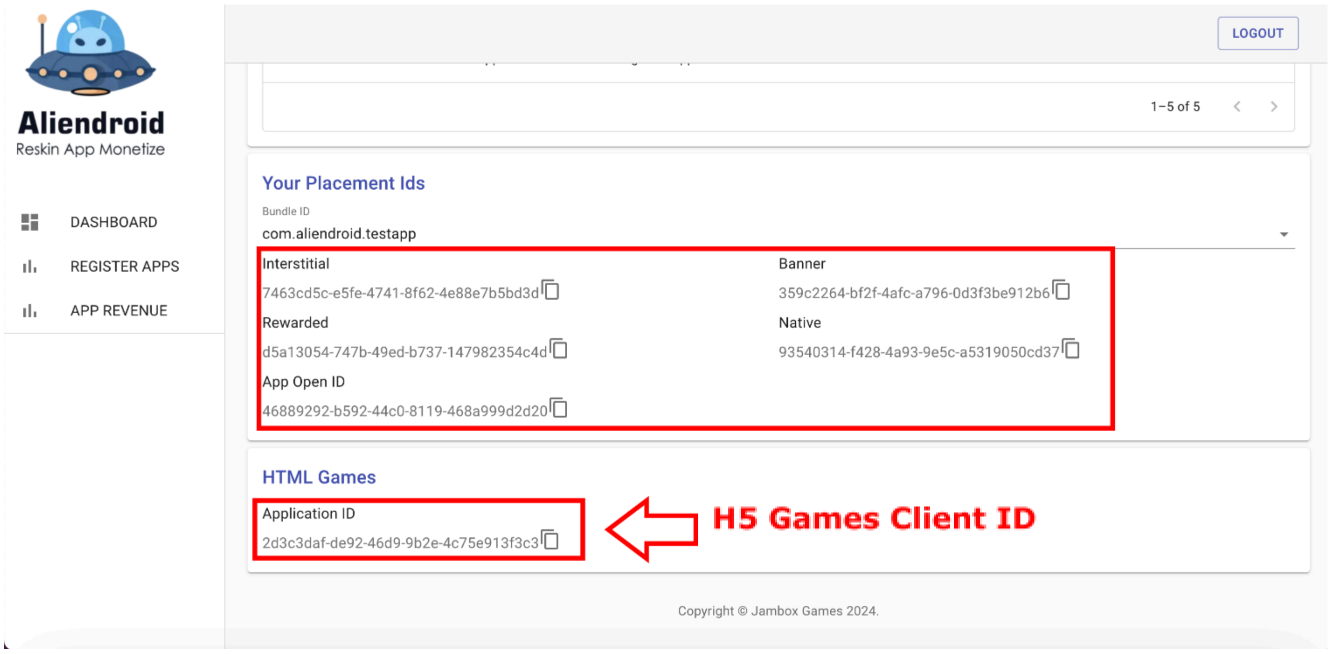
Dashboard Setup:

Before integrating the SDK, make sure to register the application on the [Jambox dashboard](). Create an account, if you don't have one already.

Register your application in the "REGISTER APPS" page



Once you register the application, you can get the H5 Games Client ID and the ad unit IDs from the dashboard



Your Placement Ids

Bundle ID
com.aliendroid.testapp

Interstitial
7463cd5c-e5fe-4741-8f62-4e88e7b5bd3d

Banner
359c2264-bf2f-4afc-a796-0d3f3be912b6

Rewarded
d5a13054-747b-49ed-b737-147982354c4d

Native
93540314-f428-4a93-9e5c-a5319050cd37

App Open ID
46889292-b592-44c0-8119-468a999d2d20

HTML Games

Application ID
2d3c3daf-de92-46d9-9b2e-4c75e913f3c3

Importing the SDK:

Import the SDK by adding the path of the AAR file in the dependencies section of *build.gradle(app)*. The SDK uses Applovin to show ads, so import Applovin SDK along with this.

```
dependencies {
        .....
        implementation(files("C:\\...\\JamboxSDK.aar"))
        implementation("com.applovin:applovin-sdk:12.4.0")
}
```

Initializing the Ads:

Make sure to initialize the Ads before using the H5 games, because the H5 games depends on it. Add the Applovin SDK key *<meta-data>* element to the AndroidManifest inside the *<application>* element.
Applovin SDK Key:
T7PPns0K6JV00uGv0ZAEKsTWrpwA-N4Hchi_KKecaqTa_U5zQcyyoI_pTcC5TM1OgfrLz5dWGdASKWgK6l5Sks

```
<manifest>
    <application>
        .......
        <!—Add the following metadata to add the SDK key -->
        <!-- Replace APPLOVIN_SDK_KEY with the key given above -->
        <meta-data
            android:name="applovin.sdk.key"
            android:value="APPLOVIN_SDK_KEY"/>
        .......
    </application>
</manifest>
```

In your MainActivity, import the *com.jambox.monetisation.JamboxAdsHelper* package, so that *JamboxAdsHelper* can be used.

```
import com.jambox.monetisation.JamboxAdsHelper;
```

Call the *InitializeAds()* method of *JamboxAdsHelper* to initialize the ads. INTERSTITIAL, REWARDED and BANNER ads are required for the H5 games to work properly, so these ad formats are initialized automatically when you call this method.
There is also APP OPEN and NATIVE ADS, which you will have to initialize manually if you are going to use them.
You can pass in a *OnJamboxAdInitializeListener* if you want to be notified when initialization is complete.
Get the ad unit id for each format from the Jambox dashboard.

```
public class MainActivity extends AppCompatActivity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        ............
        //Get the ad unit for each format from the Jambox dashboard,
        //Pass in the context and ad unit id for each ad format
        JamboxAdsHelper.InitializeAds(CONTEXT, "INTERSTITIAL_ID", "REWARDED_ID", “BANNER_ID”);
                                                    OR
        // OnJamboxAdsInitialized  will be called when the initialization is completed
        JamboxAdsHelper.InitializeAds(CONTEXT, "INTERSTITIAL_ID", "REWARDED_ID", “BANNER_ID”,
                new OnJamboxAdInitializeListener()
                {
                        @Override
                        public void OnJamboxAdsInitialized ()
                        {
                                //Initialization is completed
                        }
                });

    }
}
```

Interstitial Ads:

Use *ShowInterstitial()* method to shown an interstitial ad. You can pass in a *OnInterstitialAdListener* if you want to listen to specific events

```
JamboxAdsHelper.ShowInterstitial(new OnInterstitialAdListener()
            {
                //Called if the ad fails to be shown for some reason
                @Override
                public void OnAdDisplayFailed() { }

                //Called as soon as the ad is shown
                @Override
                public void OnAdDisplayed() { }

                //Called when the ad is hidden after the ad is completed
                @Override
                public void OnAdHidden() { }
            });
```

If you don't want to listen to interstitial events, simple pass in a null value

```
//pass in a null value for OnInterstitialAdListener if event callbacks are not needed
JamboxAdsHelper.ShowInterstitial(null);
```

Rewarded Ads:

Use *ShowRewarded()* method to shown a rewarded ad. You can pass in a *OnRewardedAdListener* to listen to specific events

```
JamboxAdsHelper.ShowRewarded(new OnRewardedAdListener()
            {
                //Called if the rewarded ad fails to be shown for some reason
                @Override
                public void OnAdDisplayFailed() { }



                //Called as soon as the rewarded is shown
                @Override
                public void OnAdDisplayed() { }

                //Called when the rewarded ad is completed, You this to reward the user
                @Override
                public void OnAdCompleted() { }

                //Called when the ad is hidden after the ad is completed
                @Override
                public void OnAdHidden() { }
            });
```

Banner Ads:

Use *ShowBannerAd ()* method to show banner ads. You will need to provide the Context and the position *(JamboxAdsHelper.BannerPosition)*

```
//You can place the banner at the top or the bottom of the screen
JamboxAdsHelper.ShowBannerAd(CONTEXT, JamboxAdsHelper.BannerPosition.TOP);
```

You can hide the banner using *HideBannerAd()* method

```
JamboxAdsHelper.HideBannerAd();
```

App Open Ads:

You have to initialize the App Open ads manually before you can use it.
The *OnJamboxAdsInitialized* event of the *OnJamboxAdInitializeListener* is the best place to initialize the App Open Ads.

```
JamboxAdsHelper.InitializeAppOpenAds("APPOPENAD_ID");
```

Use *ShowAppOpenAd()* of *JamboxAdsHelper to show the App open ad*

```
JamboxAdsHelper.ShowAppOpenAd();
```

Native Ads:

Same as App open ads, Native ads has to the initialized manually before you can use it.
Use the *OnJamboxAdsInitialized* event of the *OnJamboxAdInitializeListener* to initialize the Native Ads.

```
JamboxAdsHelper.InitializeNativeAd("NATIVE_AD_ID");
```

There are two formats of native ads that can be shown – SMALL and MEDIUM. Use the correct format depending on your use case.
You will have to provide the *FrameLayout* in which the native ads will be shown

```
//Use NativeAdTemplate.SMALL to use the SMALL format
JamboxAdsHelper.ShowNativeAd(findViewById(R.id.native_ad_small),
                                        JamboxAdsHelper.NativeAdTemplate.SMALL);
                                            OR
//Use NativeAdTemplate.MEDIUM to use the MEDIUM format
JamboxAdsHelper.ShowNativeAd(findViewById(R.id.native_ad_small),
                                        JamboxAdsHelper.NativeAdTemplate. MEDIUM);
```

To hide the native ads, use the *HideNativeAd()* method, and make sure to remove all the views from the *FrameLayout* you provided for the native ad

```
//Removing the views from the FrameLayout
```

```
((FrameLayout)findViewById(R.id.native_ad_small)).removeAllViews();
((FrameLayout)findViewById(R.id.native_ad_medium)).removeAllViews();

//Destroying the native ad
JamboxAdsHelper.HideNativeAd();
```

H5 Games:

Import *com.jambox.monetisation.WebviewObject* package so that you can enable the H5 Games.

```
import com.jambox.monetisation.WebviewObject;
```

The *JamboxAdsHelper* has to be initialized before starting the H5 games for it to work properly. Use the *IsInitialized* property inside the *JamboxAdsHelper* to make sure *JamboxAdsHelper* is initialized before starting H5 games.
To start H5 games, create an instance of *WebviewObject* by passing in the CONTEXT and H5 Games Client ID. Call ***StartWebview****()* to start H5 Games.

You need to pass the "**H5_CLIENT_ID**" from the dashboard for tracking revenue generated from your application.

```
//Create an instance by passing in the context and the client id for H5 Games
WebviewObject webviewObject = new WebviewObject(CONTEXT, "H5_CLIENT_ID");
//Calling the StartWebview() to start H5 games
webviewObject.StartWebview();
```

You can close the H5 games by using *CloseWebview()* method of the *WebviewObject*

```
//Make sure you store a reference to the WebviewObject while creating it, so you can use it here
webviewObject.CloseWebview();
```