Google AdMob Kalipso Sample

1. Components

a. **AdMobSample** – Kalipso project that contains Global Actions Sets to interface with Google AdMob
b. **AdMobSample.apk** – Compiled APK for the AdMobSample project
c. **KAdMobPlugin** – Android Studio source code that exposes the AdMob API to Kalipso

2. Requirements

a. Android device with Internet Connection
b. Kalipso Designer
c. Registered account with Google AdMob

3. How to use AdMob with Kalipso

You can use this sample to test Google AdMob for showing Ads. The provided APK is ready to install and test. This is only for Android, and in order to run, you need to compile your own APK, it can not be used with KClient on the device, since KClient does not contain the necessary Manifest Tags for Google AdMob.

First, you need to register with AdMob to get your account ID.

You should also check out AdMob documentation to understand the different types of Ad components:

https://developers.google.com/admob/android/quick-start

When you generate an APK with Kalipso you need to provide your ID in the manifest. If you open the provided Kalipso project, and generate the APL, in the last Tab (Manifest), in the “Application addons” field, there is a sample of the necessary values.
In the “meta-data” tag in the “android:value” we proved a sample of the AdMob ID, but you should replace with your own. If you create a new Kalipso project, when you build the APK, you need to fill in this in the manifest of that project.

Then, if you link to the provided Kalipso project, you can invoke the Global Action Sets to use Google AdMob in your project. You will receive “App Notified” events in your forms to notify you of events in the API. Check our sample project to see how it works.

- **asAdMob_Initialize** – Should be called once in your project to initialize the AdMob API. When the API is initialized your App will be notified with “App Notified” event and in TVAR(0) will contain "AdMob_Initialized". After this, you can call the other Action Sets.

- **asAdMob_UnInitialize** – Should be called when you no longer need to use AdMob, normally before closing your project.

- **asAdMob_AddBanner** – Can be called to show an Ad Banner in your App.
  First you should create a Kalipso control in your form to be used as a placeholder. For example, you can create an invisible Label on your form, and pass the control name to this Action Set, to should an Ad Banner in the place where the Label is.

- **asAdMob_RemoveBanner** – Can be called to remove an Ad Banner previously added with a call to asAdMob_AddBanner.

- **asAdMob_LoadInterstitial** – Can be called for the Ad API to load an Ad that will be shown covering the full screen. This call only prepares the Ad, then you need to wait for an “App Notified” event with TVAR(0) containing "AdMob_AdLoaded". When you receive this notification, the Ad is ready can be displayed at any moment with a call to asAdMob_ShowInterstitial.

- **asAdMob_ShowInterstitial** – When you received the "AdMob_AdLoaded" notification, this can be called to show the Ad to the user.

- **asAdMob_LoadReward** - Can be called for the Ad API to load an Ad that will be shown covering the full screen, normally with a video, that if the user sees until the end, will give him a reward. This call only prepares the Ad, then you need to wait for an “App Notified” event with TVAR(0) containing "AdMob_AdLoaded". When you receive this notification, the Ad is ready can be displayed at any moment with a call to asAdMob_ShowReward.

- **asAdMob_ShowReward** - When you received the "AdMob_AdLoaded" notification, this can be called to show the Ad to the user. If the user sees the Ad until the end and should receive a reward, you will receive an “App Notified” event with TVAR(0) containing "AdMob_UserEarnedReward", and in TVAR(2) the reward amount.

These Action Sets receive an “Unit ID”, that is your Ad Unit ID registered in your AdMob Console. Read the AdMob documentation (link provided above) for more details. When you are testing however, you should leave this value empty, and Google Test Unit ID will be used, or provide Googles Test Unit IDs yourself. You should not use real production Unit IDs for testing, or you can be banned from AdMob for “fake” Ad usage. Once more, **read Google AdMob documentation**.

They also receive an “Internal ID” that will be sent back to you in TVAR(1) whenever “App Notified” events are returned to you, in case you use different Ad modes and you need to know to what Ad they correspond.

4. Further extend the connection between Kalipso and HD4000

In folder KAdMobPlugin, is the Android Studio source project used to expose the AdMob API to Kalipso. You can modify it, or create your own, to add additional capabilities to the Kalipso connection. You should then compile a KAdMobInterface.apk.

We have already done this in this sample, and this compiled APK has been copied to the FilesToSend folder in the AdMobSample. If you modify this project and compile a new APK, you should put the new APK in the FilesToSend folder.