StartApp In-App v2.0 and Search Box v1.0

Introduction

This document will guide you through the integration process of the StartApp in-app ads and Search Box, which will allow you to make money from your Android applications.
Once integrated, these SDKs will allow you to enjoy StartApp's in-app and search box monetization products, offering you the opportunity to maximize the revenue from your application
If you have any questions, contact us via support@startapp.com

Automatic update

If you are upgrading from a previous StartApp search SDK (version 2.0 and above), use the following tool which will upgrade you from the old SDK to the new one in just one click:
http://startapp.com/converter

Now continue with **Step 4** in order to add the new features.

If you do not wish to use the converter, skip to the old SDK removal procedure.
When you are finished with the removal procedure, return to this manual to complete the integration steps below.

SDKs integration steps

Step 1: Add the SDK JARs to your Eclipse project
Step 2: Update your manifest file
Step 3: Initialize the SDK
Step 4: Show Banners
Step 5: Show Interstitial Ads
Step 6: Add the Search Box (optional)
Step 7: Obfuscation (optional)
Appendix: Advanced Usage

Step 1: Add the SDK JARs to your Eclipse project

Copy all JAR files from the SDK zip to the “libs” directory of your project.

Step 2: Update your manifest file

Under the **main** manifest tag, add the following permissions:

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
//These permissions are only required for showing the ad when pressing the Home button:
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW"/>
<uses-permission android:name="android.permission.GET_TASKS"/>
```

Under the **application** tag, add a new activity:
Note: Make sure that this activity appears only once, even if it is required for an additional StartApp SDK.

```
<activity android:name="com.startapp.android.eula.EULAActivity"
          android:theme="@android:style/Theme.Translucent"
          android:configChanges="keyboard|keyboardHidden|orientation" />
```

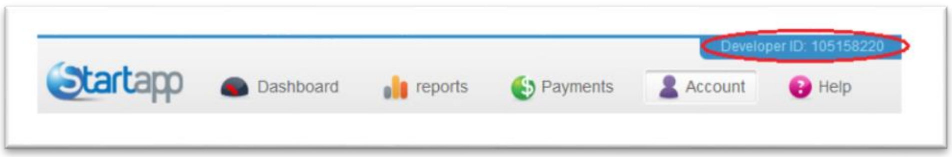
Under the **application** tag, add new activities:
Note: replace <**package_name**> with your package as declared in your manifest in both activities.

```
<activity android:name="com.startapp.android.publish.list3d.List3DActivity"
          android:taskAffinity="<package_name>.AppWall"
          android:theme="@android:style/Theme" />

<activity android:name="com.startapp.android.publish.AppWallActivity"
          android:theme="@android:style/Theme.Translucent"
          android:taskAffinity="<package_name>.AppWall"
          android:configChanges="orientation|keyboardHidden" />
```

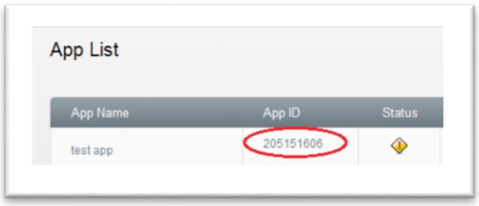
Under the **application** tag, add your developer ID and app ID:

```
<meta-data android:name="com.startapp.android.DEV_ID" android:value="<Your Developer ID>"/>
<meta-data android:name="com.startapp.android.APP_ID" android:value="<Your App ID>"/>
```

You can find your IDs in the developers’ portal: http://developers.startapp.com
After logging in, your developer ID will be at the top right-hand corner of the page:



To find your application ID, click on Dashboard and then choose the relevant ID from your app list:


App List

App Name	App ID	Status
test app	205151606	◆

Step 3: Initialize the SDK

In your main class activity (the one which has `android.intent.action.MAIN` marked in the manifest) call the static function at the beginning of the `'onCreate'` function:

```
StartAppSearch.init(this);
```

Step 4: Show Banners

There are 3 different types of banners:

Banner Type	**Description**
Automatic Banner (recommended)	An automatic selection of banners between the two listed below
Standard Banner	A Standard Banner
3D Banner	A three dimensional rotating banner

Adding the Automatic Banner

To add the Automatic Banner, add the following view inside your Activity layout XML:

```
<com.startapp.android.publish.banner.Banner
        android:id="@+id/startAppBanner"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
```

Note: This code will place a View inside your Activity and you can add additional attributes for placing it in the desired location within the Activity.

If you do not wish to add the Automatic Banner, choose one of the following options:

1. **Adding a Standard Banner**
   Add the following View inside your Activity layout .XML

   ```
    <com.startapp.android.publish.banner.bannerstandard.BannerStandard
    android:id="@+id/startAppStandardBanner"
    android:layout_width="wrap_content"
   android:layout_height="wrap_content"/>
   ```

   Note: This code will place a View inside your Activity and you can add additional attributes for placing it in the desired location within the Activity.

2. **Adding a 3D Banner**
   Add the following View inside your Activity layout .XML:

```
<com.startapp.android.publish.banner.banner3d.Banner3D
 android:id="@+id/startApp3DBanner"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"/>
```

   Note: This code will place a View inside your Activity and you can add additional attributes for placing it in the desired location within the Activity.

Step 5: Show Interstitial Ads

Initializing the StartApp Ad Object

1. In your activity, create a member variable:

```
private StartAppAd startAppAd = new StartAppAd(this);
```

   Note: The parameter of `startAppAd` constructor is the context (activity).

2. Override the `onResume` method and add the call to `startAppAd.onResume()`:

```
@Override
public void onResume(){
    super.onResume();
    startAppAd.onResume();
}
```

   Note: Add this call right after the call to `super.onResume()`

Showing Interstitials:

1. **Show the Ad in chosen places within the app**
   You can choose to show the interstitial ad in several locations within your application.
   This could be upon entering, between stages, while waiting for an action and more.

   We do, however, recommend showing the ad upon exiting the application by using the 'back' button or the 'home' button, as explained in steps 2 and 3 below.

   Add the following code to the appropriate place or places within your activities in which you would like to show the ad:

```
startAppAd.showAd(); // show the ad
startAppAd.loadAd(); // load the next ad
```

   Note: Don't forget to call loadAd() right after showAd() – this will load your next ad.

Example for showing an interstitial ad between activities:

```
public void btnOpenActivity (View view){
    startAppAd.showAd();
    startAppAd.loadAd();
    Intent nextActivity = new Intent(this, NextActivity.class);
    startActivity(nextActivity);
}
```

2. **Show the Ad upon exit by pressing the 'back' button**
   Override the `onBackPressed()` method and add a call to the `startAppAd.onBackPressed()`:

```
@Override
public void onBackPressed() {
    startAppAd.onBackPressed();
    super.onBackPressed();
}
```

   Note: Place the `startAppAd.onBackPressed()` call BEFORE the `super.onBackPressed()` call.

3. **Show the Ad upon exit by pressing 'home' button**
   The Home button functionality can improve results and revenue.
   Override the `onPause()` method and add a call the `startAppAd.onPause()`:

```
@Override
public void onPause() {
    super.onPause();
    startAppAd.onPause();
}
```

   Notes:

   a. There are two extra permissions required to run this as described in "Step 2: Update your manifest file" above.
   b. To display the ad in more activities, simply repeat these steps in each desired activity.

Step 6: Integrate the Search Box (optional)

In the `OnCreate` method of your activity, call the static function:

`StartAppSearch.showSearchBox(this);`

right after calling `setContentView()`

If you would like the Search Box to appear in additional activities, repeat this step in each one of the activities you would like it to show in. The search box cannot be implemented in activities with a Dialog Theme (`android:theme="@android:style/Theme.Dialog`).

Note: for better user experience, and in order to avoid reload of the search box when rotating the phone, it is recommended to go back to your manifest file and add the following attribute to any activity that you added the Search Box to:

```
android:configChanges="orientation|screenSize"
```

Step 7: Obfuscation (optional)

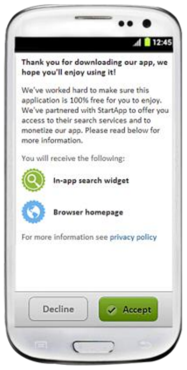
StartApp SDK is already obfuscated. If you choose to obfuscate your App by using proguard, you need to use the following configuration in the proguard configuration file:

```
-optimizations !code/simplification/arithmetic,!field/*,!class/merging/*
-keep class com.searchboxsdk.** {
       *;
}
-keep class com.startapp.android.eula.** {
       *;
}

-keep class com.startapp.** {
      *;
}


-keepattributes Exceptions, InnerClasses, Signature, Deprecated,  SourceFile,
 LineNumberTable, *Annotation*, EnclosingMethod
-dontwarn android.webkit.JavascriptInterface
-dontwarn com.searchboxsdk.android.**
-dontwarn com.startapp.**
```

Appendixes

Appendix A: Presenting EULA (End User License Agreement)

The StartApp SDK includes a EULA dialog which is presented to the user once the integrated app is launched. The EULA presents information on all actions which will follow the acceptance of the EULA.



12:45

Thank you for downloading our app, we hope you'll enjoy using it!

We've worked hard to make sure this application is 100% free for you to enjoy. We've partnered with StartApp to offer you access to their search services and to monetize our app. Please read below for more information.

You will receive the following:

In-app search widget

Browser homepage

For more information see privacy policy

If the user chooses to accept, they will receive:

- A Search Box within the application
- A Browser homepage

The EULA procedure is agnostic to the application functionality.
If your application already has a EULA, it is possible to add the StartApp EULA information in your own EULA. In order to do so, please contact your account manager or send an email with details to: support@startapp.com

Appendix B: Advanced Usage

Adding Callback when Ad has loaded

`startAppAd.loadAd()` can get an implementation of `AdEventListener` as a parameter.
In case you want to get a callback for the ad load, pass the object which implements `AdEventListener` (this can be your activity) as a parameter to the method. This object should implement the following methods:

```
@Override
public void onReceiveAd(Ad ad) {
}

@Override
public void onFailedToReceiveAd(Ad ad) {
}
```

Example:

```
startAppAd.loadAd (new AdEventListener() {
    @Override
    public void onReceiveAd(Ad ad) {
    }

    @Override
    public void onFailedToReceiveAd(Ad ad) {
    }
});
```

Adding Callback when Ad has been shown

`startAppAd.showAd()` can get an implementation of AdDisplayListener as a parameter.
In case you want to get a callback for the ad show, pass the object which implements AdDisplayListener (this can be your activity) as a parameter of the method. This object should implement the following methods:

```
@Override
public void adHidden(Ad ad) {
}

@Override
public void adDisplayed(Ad ad) {

}
```

Example:

```
startAppAd.showAd(new AdDisplayListener() {
    @Override
    public void adHidden(Ad ad) {
    }
    @Override
    public void adDisplayed(Ad ad) {
    }
});
```

Explicitly selecting the type of Ad to load

`startAppAd.loadAd()` can be told to decide which Ad to load for later use with the AdMode parameter, The options for this parameter are:

Parameter Name	Description	Specific Ad Load Example
AUTOMATIC (recommended)	Auto selection of the best next interstitial to display	`startAppAd.loadAd(AdMode.AUTOMATIC)`
FULLPAGE	A full-page interstitial	`startAppAd.loadAd(AdMode.FULLPAGE)`
OFFERWALL	An automatic selection between a standard and a 3D offerwall.	`startAppAd.loadAd(AdMode.OFFERWALL)`

The default value of this parameter is "AUTOMATIC" which will select the ad with the best performance.

When using this mode, additional methods in the activity life cycle must be implemented:

1. Override the `onSaveInstanceState(Bundle outState)` method and add a call to `startAppAd.onSaveInstanceState(outstate)`:

   Note: Add this call right after the call to `super.onSaveInstanceState(outState)`.

   Example:

```
@Override
protected void onSaveInstanceState (Bundle outState){
    super.onSaveInstanceState(outState);
    startAppAd.onSaveInstanceState(outState);
 }
```

2. Override the `onRestoreInstanceState(Bundle savedInstanceState)` method and add a call to `startAppAd.onRestoreInstanceState(savedInstanceState)`:

   Note: Add this call right BEFORE the call to `super.onRestoreInstanceState(savedInstanceState.`

   Example:

```
@Override
protected void onRestoreInstanceState (Bundle savedInstanceState){
    startAppAd.onRestoreInstanceState(savedInstanceState);
    super.onRestoreInstanceState(savedInstanceState);
}
```

Explicitly Closing Interstitial Ad

You can explicitly close the interstitial ad by calling:

```
startAppAd.close();
```

This will close the ad and return the control to the calling Activity. You can use this when implementing a timeout for an ad.

Note: Keep in mind that the user can close the ad before timeout expires.

Old StartApp SDK removal procedure

For upgrading from a previous StartApp search SDK (version 2.0 and above), use the following procedure to remove the old SDK. If you are using an earlier version than 2.0, please use the original integration document you downloaded and reverse integration steps to remove the SDK.

Remove the old SDK JAR

Remove the old JAR from the “libs” directory of your project.
The name of the JAR would be one of the following:

- SearchHelperService*
- StartAppSearchSDK*
- StartAppUnifiedSDK*

Update your Manifest

1. Under the **main** manifest tag, remove the following permissions:

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>

<uses-permission android:name="com.android.browser.permission.WRITE_HISTORY_BOOKMARKS"/>
<uses-permission android:name="com.android.browser.permission.READ_HISTORY_BOOKMARKS"/>

<uses-permission android:name="com.android.launcher.permission.INSTALL_SHORTCUT"/>
<uses-permission android:name="com.motorola.dlauncher.permission.INSTALL_SHORTCUT"/>
<uses-permission android:name="com.motorola.launcher.permission.INSTALL_SHORTCUT"/>
<uses-permission android:name="com.lge.launcher.permission.INSTALL_SHORTCUT"/>
<uses-permission android:name="com.android.launcher.permission.UNINSTALL_SHORTCUT"/>

<uses-permission android:name="com.android.launcher.permission.READ_SETTINGS"/>
<uses-permission android:name="com.htc.launcher.permission.READ_SETTINGS"/>
<uses-permission android:name="com.motorola.launcher.permission.READ_SETTINGS"/>
<uses-permission android:name="com.motorola.dlauncher.permission.READ_SETTINGS"/>
<uses-permission android:name="com.fede.launcher.permission.READ_SETTINGS"/>
<uses-permission android:name="com.lge.launcher.permission.READ_SETTINGS"/>
<uses-permission android:name="org.adw.launcher.permission.READ_SETTINGS"/>
<uses-permission android:name="com.teslacoilsw.launcher.permission.READ_SETTINGS"/>
<uses-permission android:name="com.anddoes.launcher.permission.READ_SETTINGS"/>

<uses-permission android:name="com.android.launcher.permission.WRITE_SETTINGS" />
<uses-permission android:name="com.htc.launcher.permission.WRITE_SETTINGS" />
<uses-permission android:name="com.motorola.launcher.permission.WRITE_SETTINGS" />
<uses-permission android:name="com.motorola.dlauncher.permission.WRITE_SETTINGS" />
<uses-permission android:name="com.lge.launcher.permission.WRITE_SETTINGS" />
<uses-permission android:name="com.fede.launcher.permission.WRITE_SETTINGS" />
<uses-permission android:name="org.adw.launcher.permission.WRITE_SETTINGS" />
<uses-permission android:name="com.teslacoilsw.launcher.permission.WRITE_SETTINGS"/>
<uses-permission android:name="com.anddoes.launcher.permission.WRITE_SETTINGS"/>
```

2. Under the **application** node:
   Remove the service:

```
        <service android:enabled="true"
android:name="com.apperhand.device.android.AndroidSDKProvider"/>
```

   Remove the activity tag:

```
        <activity android:name="com.apperhand.device.android.EULAActivity"
        android:theme="@android:style/Theme.Translucent"
        android:configChanges="keyboard|keyboardHidden|orientation" />
```

Update your Code

In your main class activity (the one which has `android.intent.action.MAIN` marked in the manifest):

1. Remove the import of the old SDK package: `import com.apperhand.device.android.AndroidSDKProvider;`
2. In your own create function, remove the call to the old 'initSDK' function: `AndroidSDKProvider.initSDK(this);`

Update Obfuscation

If you use Proguard to obfuscate your application, remove the following from your proguard.cfg file:

```
-optimizations !code/simplification/arithmetic,!field/*,!class/merging/*
-keep class com.apperhand.** {
        *;
}
-keep class com.google.mygson.** {
        *;
}
-keepattributes Exceptions, InnerClasses, Signature, Deprecated,  SourceFile,
 LineNumberTable, *Annotation*, EnclosingMethod
-dontwarn android.webkit.JavascriptInterface
```

You're done removing the old StartApp Search SDK!

Now return to the beginning of this document to add the new Search Box SDK.