OOYALA®

GETTING STARTED GUIDE:
ANDROID SDK

Last Updated: January 29, 2013

INTRODUCTION

The Ooyala Android SDK provides you with a set of tools to rapidly develop rich, custom video application experiences for Android devices. The SDK includes APIs to display custom playlists, play video, display VAST and Ooyala ads, and more.

This guide describes how to get started with the Ooyala Android SDK.

DEVELOPMENT REQUIREMENTS

Before you begin, make sure you have the following:

- The Ooyala Android SDK and sample application (OoyalaSDK.zip).
- An Android development environment (http://developer.android.com/sdk).

SUPPORTED DEVICES

The Android SDK supports devices running Android OS Version 2.2 or later.

SUPPORTED DELIVERY METHODS

The Android SDK supports the following delivery methods:

- HTTP Live Streaming (HLS) for Android 4.0 or later
- MP4

GETTING STARTED

To get started:

1. Locate the OoyalaSDK.jar from within the OoyalaSDK.zip and choose an installation location.
2. Open Eclipse.
3. Click **File**, select **New**, and click **Android Project**. If **Android Project** does not appear, Click **File**, select **New**, and click **Other**. Then, expand **Android** and click **Android Project**. The **New Android Project** dialog box appears.
4. Follow the steps to create a new project. Eclipse creates a new Android project.
5. Add the OoyalaSDK.jar to your Android application. For example, you might do the following:
   a. Create a lib folder by selecting **File**, pointing to **New**, and clicking **Folder**.
   b. Dragging the OoyalaSDK.jar to the lib folder.
6. Add the OoyalaSDK.jar to the classpath of your Android application. For example, you might do the following:
   a. Click **Project** and select **Properties**.
   b. Select **Java Build Path** in the left pane.
   c. Click **Add Jars**.
   d. Navigate to and select the OoyalaSDK.jar file.
   e. Click **OK**.
7. Select the AndroidManifest.xml file and do the following:
   a. Click the **Permissions** tab.
   b. Click **Add**.
   c. Select **Uses Permission** and click **OK**.
   d. From the **Name** list box, select android.permission.INTERNET.
   e. Click the AndroidManifest.xml tab to verify that the following entry is added:
      <uses-permission android:name="android.permission.INTERNET"/>
8. To prevent the video from restarting when the device changes orientation (vertical to horizontal or horizontal to vertical), do the following:
   a. Select the AndroidManifest.xml file.
   b. Add the following attribute to the **activity** element:
      android:configChanges="orientation|keyboardHidden"
      For example:
      <activity
        android:name=".TestOoyalaSDKActivity"
        android:label="@string/app_name"
        android:configChanges="orientation|keyboardHidden" >
   c. Ensure the application will work with Android Version 2.2 and above by changing:
      <uses-sdk android:minSdkVersion="15" />
      to:
      <uses-sdk android:minSdkVersion="8" />
9. Double-click <*project*>/res/layout/main.
10. Select the **main.xml** tab.

11. Add the following within the **LinearLayout** element:
```
<com.ooyala.android.OoyalaPlayerLayout
  android:id="@+id/ooyalaPlayer"
  android:layout_width="match_parent"
  android:layout_height="match_parent">
</com.ooyala.android.OoyalaPlayerLayout>
```
12. Double-click <*project*>/src/<*package_name*>/<*activity_source_file*>.
13. Add the following imports:
```
import android.util.Log;
import com.ooyala.android.OoyalaPlayer;
import com.ooyala.android.OoyalaPlayerLayout;
import com.ooyala.android.OoyalaPlayerLayoutController;
```
14. Paste the following beneath **setContentView**:
```
OoyalaPlayerLayout playerLayout = (OoyalaPlayerLayout)findViewById(R.id.ooyalaPlayer);
OoyalaPlayerLayoutController playerLayoutController = new OoyalaPlayerLayoutController(playerLayout, "<your api key>", "<your secret>", "<your pcode>", "<your domain>");
OoyalaPlayer player = playerLayoutController.getPlayer();
if (player.setEmbedCode("<the embed code to play>")) {
  // The Embed Code works
  player.play();
} else {
  Log.d(this.getClass().getName(), "Something Went Wrong!");
}
```
15. Replace <*your api key*>, <*your secret*>, <*your pcode*>, <*your domain*> with your API key, secret key, provider code, and hosted_at URL. To get this information, open Backlot, click the Account tab, and click the Developers subtab.
16. Replace <*the embed code to play*> with the embed code you want to play.

Note: If you have problems viewing a video, make sure the syndication settings are set to allow Android devices. For more information, go to the Publishing Rules section of the Ooyala Backlot User Guide.

WORKING WITH THE StreamSelector

Here are the basic steps for working with the `StreamSelector` interface.

1. First, you need to implement StreamSelector interface. Here is the default implementation for reference.

```
private static class DefaultStreamSelector implements StreamSelector {
    public DefaultStreamSelector() {}
    @Override
    public Stream bestStream(Set<Stream> streams) {
      if (streams == null || streams.size() == 0) { return null; }
      Stream lowestBitrateStream = null;
      for (Stream stream : streams) {
        // for remote assets, just pick the first stream
        if (stream.getDeliveryType().equals(Constants.DELIVERY_TYPE_REMOTE_ASSET)
            || stream.getDeliveryType().equals(Constants.DELIVERY_TYPE_HLS)) { return stream; }
        if (Stream.isDeliveryTypePlayable(stream)
            && (lowestBitrateStream == null
                || stream.getCombinedBitrate() < lowestBitrateStream.getCombinedBitrate()
                || (stream.getCombinedBitrate() == lowestBitrateStream.getCombinedBitrate()
            && stream.getHeight() < lowestBitrateStream.getHeight()))) {
          lowestBitrateStream = stream;
        }
      }
      return lowestBitrateStream;
    }
  }
```

2. Now, you need to force the Ooyala Player to use your `StreamSelector`. Call `Stream.setStreamSelector()` before any embed codes are set. Here is the method signature, which is defined in the `Stream` class.

```
/**
   * This method will set the StreamSelector used to select the Stream to play.
   * @param selector an implemented StreamSelector
   */
  public static void setStreamSelector(StreamSelector selector) {
    _selector = selector;
  }
```