Android Project Setup

Connected Sentinel Player SDK 2.0

viaccess·orca

copyright

trademarks

Document reference number: 21828

Document version number: 1.0 Draft

Contents

Introduction

The Connected Sentinel Player SDK Android Project Setup manual explains how to set up and configure an Android project using the Viaccess-Orca Connected Sentinel Player SDK over Win32 using Eclipse Helios.

Target Audience

The manual is aimed at software developers who need to build or extend an Android project using the Viaccess-Orca Connected Sentinel Player SDK.

Glossary

This manual contains some acronyms or terms that are specific to the field of Viaccess-Orca T. If the text does not define a term, refer to the *Glossary* on page 17 for a complete definition.

IDE Prerequisites

Android SDK

Android SDK is installed on the development machine.

The following packages from the SDK manager must be installed:

- SDK Platforms for Android 2.1 and up (e.g., Android 2.2)
- Google APIs by Google Inc. for Android API 7 and up

Eclipse IDE

Eclipse IDE Helios Version is installed on the development machine.

Android ADT Plug-in

Android ADT Plug-in is installed via Eclipse IDE.

Android Project Prerequisites

Android project must use Android 3.0 (API 11) or above.

Package Content

The Customer receives a ZIP file containing APIS JavaDocs:

- Connected Sentinel Player SDK API Reference,
- Connected Sentinel Player SDK Common Integration Guide,
- Connected Sentinel Player SDK Android Integration Guide.
- The `libs` folder containing the Connected Sentinel Player SDK and a Connected Sentinel Player API Demo source code.

The relevant material for integrating the Connected Sentinel Player SDK is contained in the `libs` folder.

Libs Folder Content

Component	Description
`DxDrmDlc.jar` File	This file (Viaccess-Orca DRM Download Client) contains the Connected Sentinel Player Java API.
`voOSBasePlayer.jar` File	This file contains a part of the software player Java API.
`voOSBaseSource.jar` File	This file contains a part of the software player Java API.
`voOSEngine.jar` File	This file contains a part of the software player Java API.
`voOSHDMICheck.jar` File	This file contains a part of the software player Java API.
`voOSUtils.jar` File	This file contains a part of the software player Java API.
`assets.jar` File	This file is required for local personalization, for debug purposes only.
`armeabi` Subfolder	This folder contains native shared objects used by the Connected Sentinel Player Java API.

Setup Procedure

Unzip Package

1. Extract the `libs` folder from the ZIP provided.
2. Verify that the structure of the folder `libs` is not changed (see Package Content - Libs Folder Content).

Create/Modify Android Project

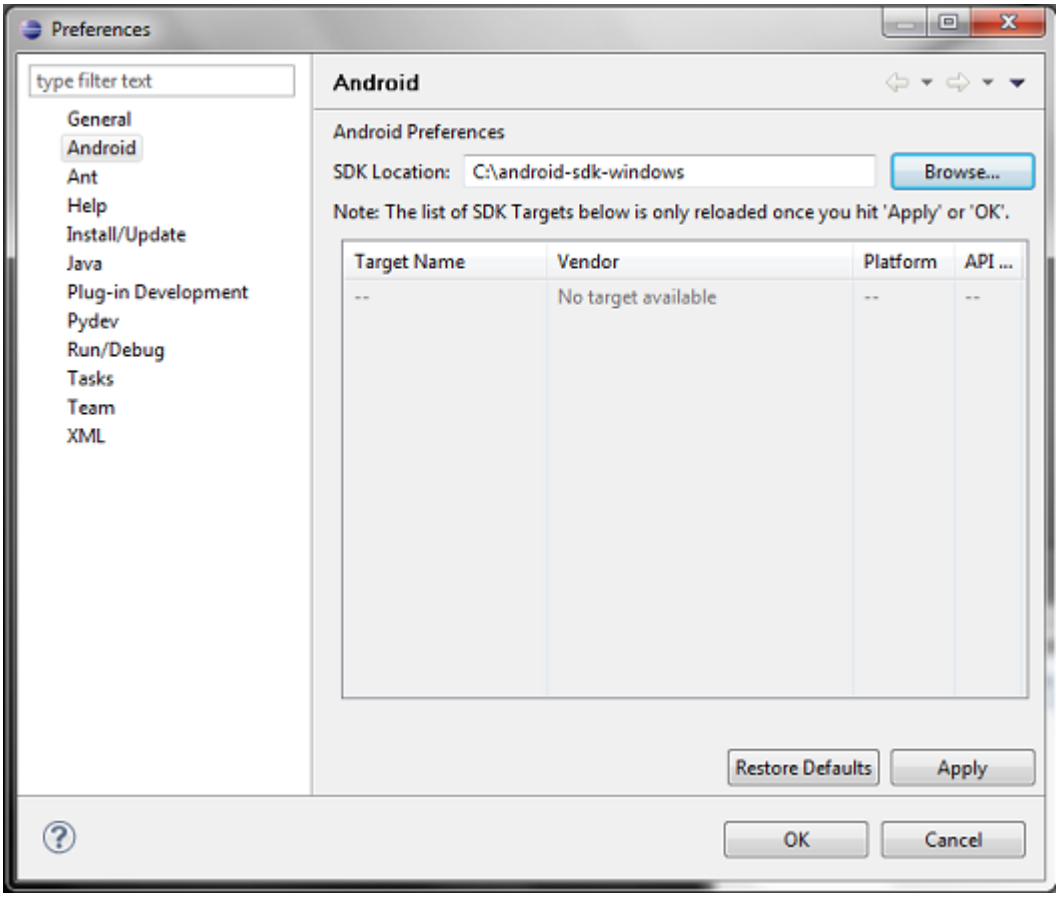
Listed below are steps to create/modify an Android Project that uses the Connected Sentinel Player API.

Create an Android Project

1. Open Eclipse Helios IDE.

Verify that the SDK Location is set. To do so, proceed as follows:

1. Click **Menu: Window > Preference**.
2. Select **Android** on the left panel.
3. Type in the path of the Android SDK within the **SDK Location** text box.



4. Click **Apply > OK**.

Open your Android project or create a new one. To create a new Android Project:

1. Click **File > New > Android Project**
2. Set:

   Project name: Sample

   Build target: Select Android 3.0

   Application name: Sample

   Package name: com.dxdrmdlc.sample

   Create Activity: Sample

   Min SDK Version: 7
3. Click **Finish**.

> **note**
>
> The values given for Application name, Package name and activity are just an example. These values may be changed according to the project specifics.

Add Connected Sentinel Player API

Create Libs Folder

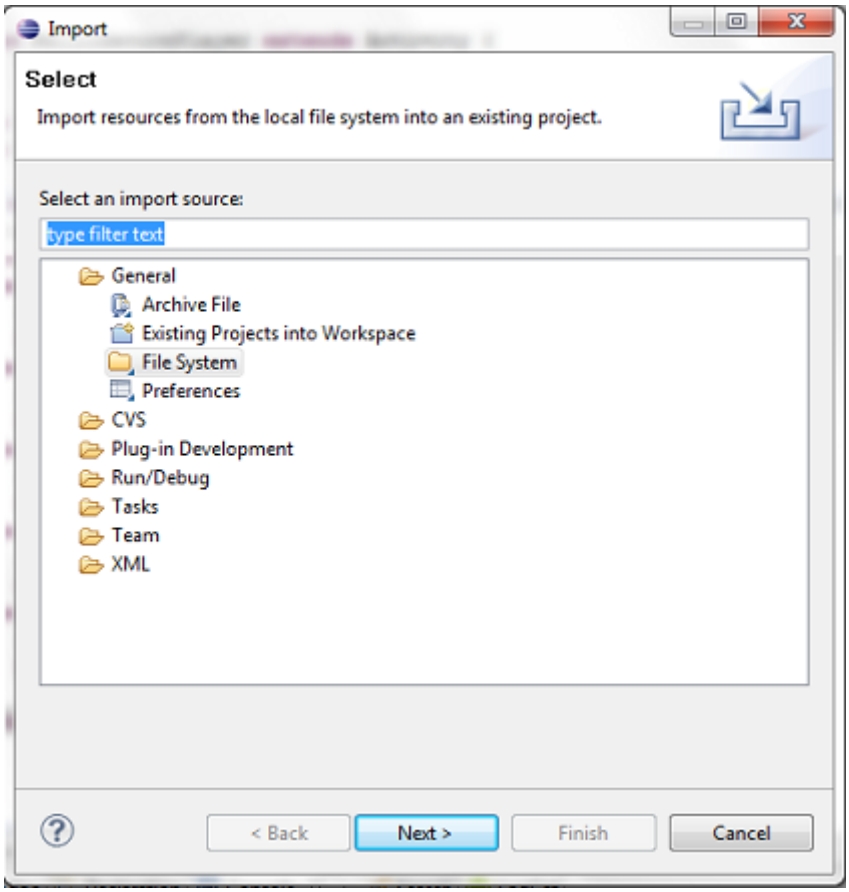
Check whether the `libs` folder exists in your project. If not, create it. To create the folder:

1. Display the **Package Explorer.**
2. Right-click the package root node (project name).
3. In the popup menu select **New > Folder.**
4. Add **Folder name**: `libs`.
5. Click **Finish**.

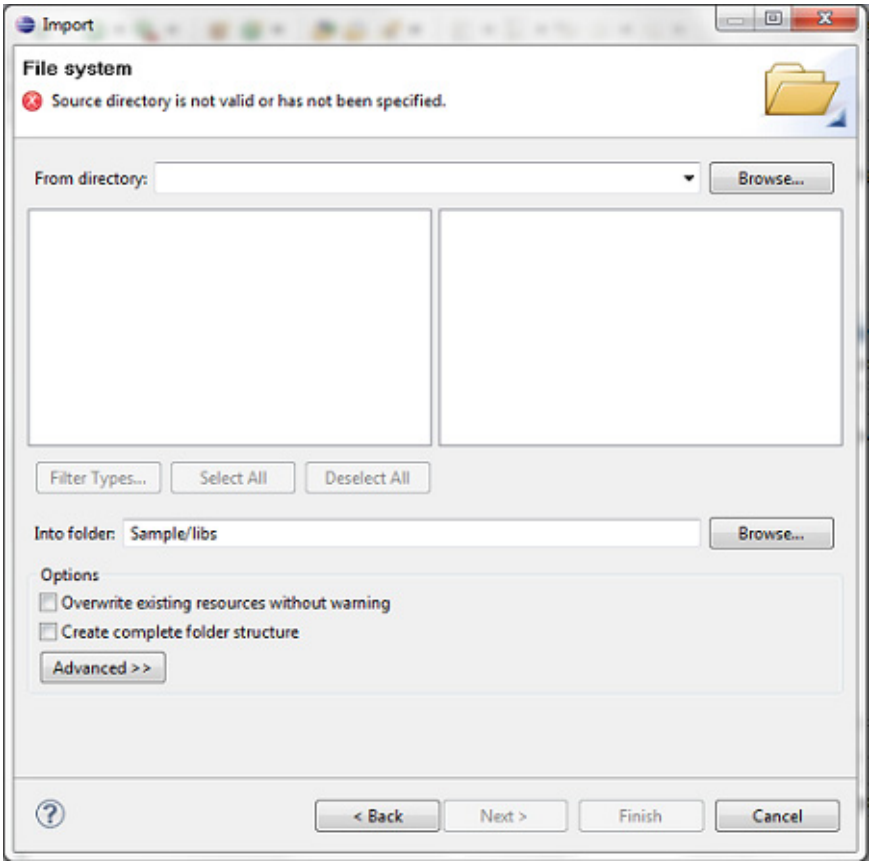
Import Jar Files

To import .Jar files:

1. Right-click the `libs` Folder.
2. In the popup menu select **Import**.
3. In the **Import** window select **General > File System**.


Import
Select
Import resources from the local file system into an existing project.
Select an import source:
type filter text
General
Archive File
Existing Projects into Workspace
File System
Preferences
CVS
Plug-in Development
Run/Debug
Tasks
Team
XML
< Back
Next >
Finish
Cancel

4. Click **Next**.

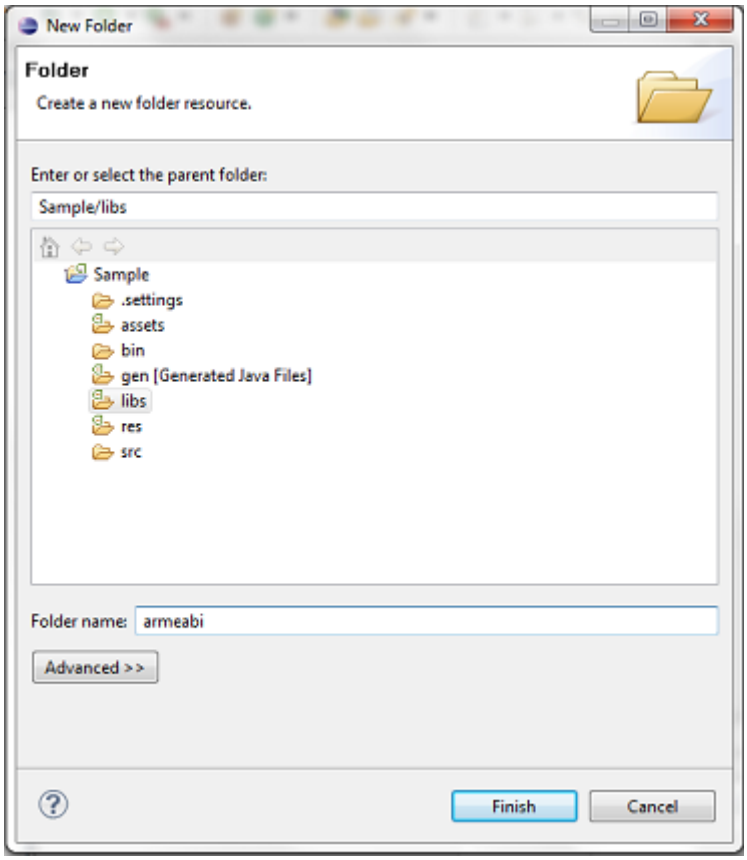

Import
File system
Source directory is not valid or has not been specified.
From directory:
Browse...
Filter Types...
Select All
Deselect All
Into folder:
Sample/libs
Browse...
Options
Overwrite existing resources without warning
Create complete folder structure
Advanced >>
< Back
Next >
Finish
Cancel

5. Browse to the location where you placed the libs folder that you unzipped from the provided Connected Sentinel Player SDK.
6. Select `assets.jar`, `DxDrmDlc.jar`, `voOSBasePlayer.jar`, `voOSDataSource.jar`, `voOSEngine.jar`, `voOSHDMICheck.jar` and `voOSUtils.jar`.
7. Click **Finish**.

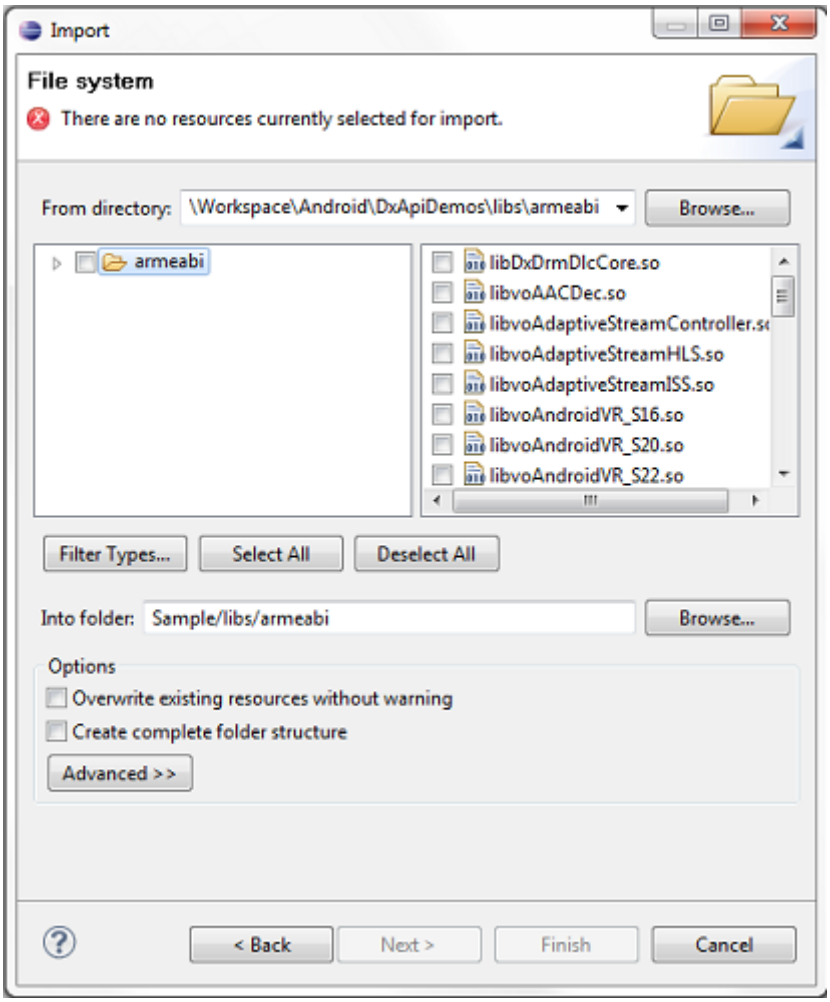
Import SO Files

To import `.so` files:

1. Right-click the `libs` Folder.
2. In the popup menu select **New > Folder**
3. Add **Folder name**: `armeabi`.



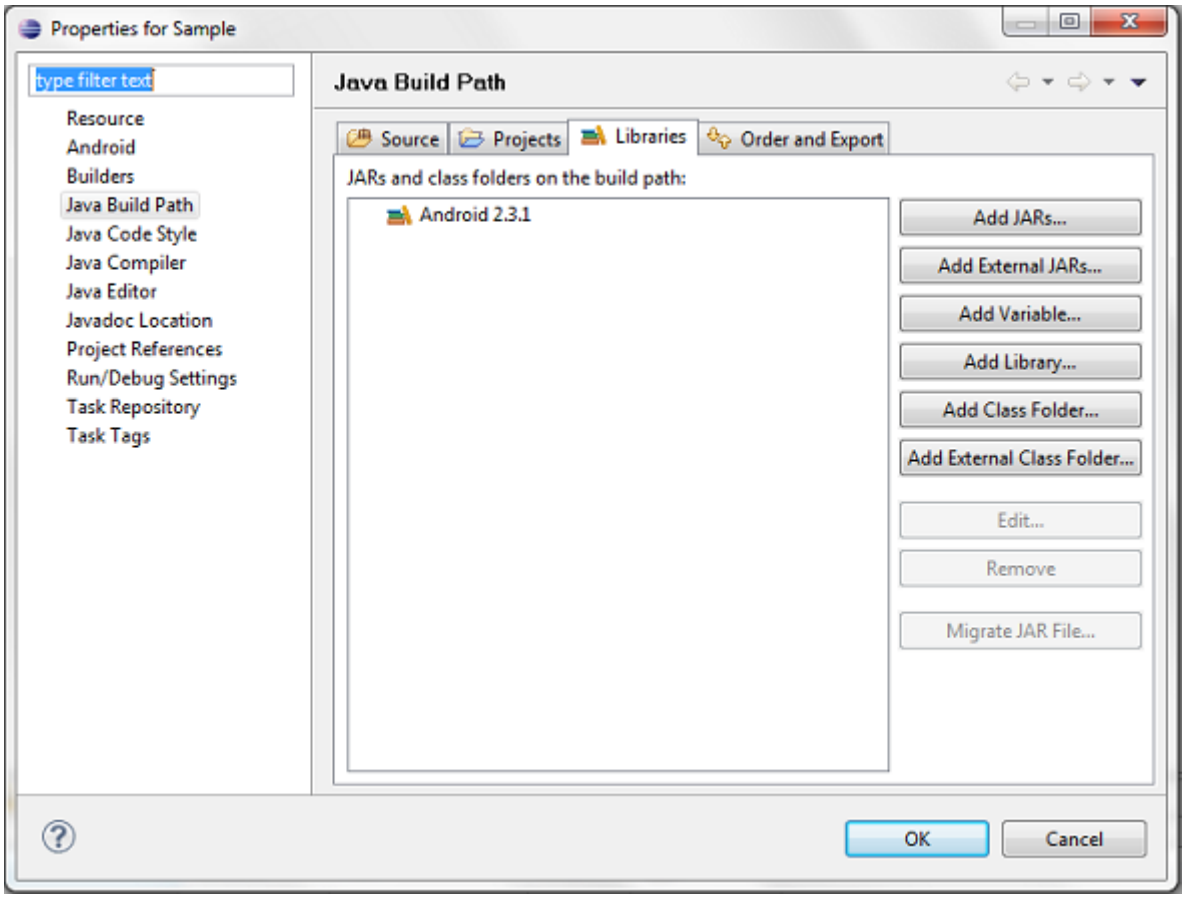
4. Click **Finish**.
5. Right-click on the `armeabi` Folder.
6. In the popup menu select **Import**
7. In the Import window select **General > File System**
8. Browse to the location where you placed the `Libs/armeabi` folder that you unzipped from the provided Connected Sentinel Player SDK.
9. Click on **Select All** (select all `*.so` files).



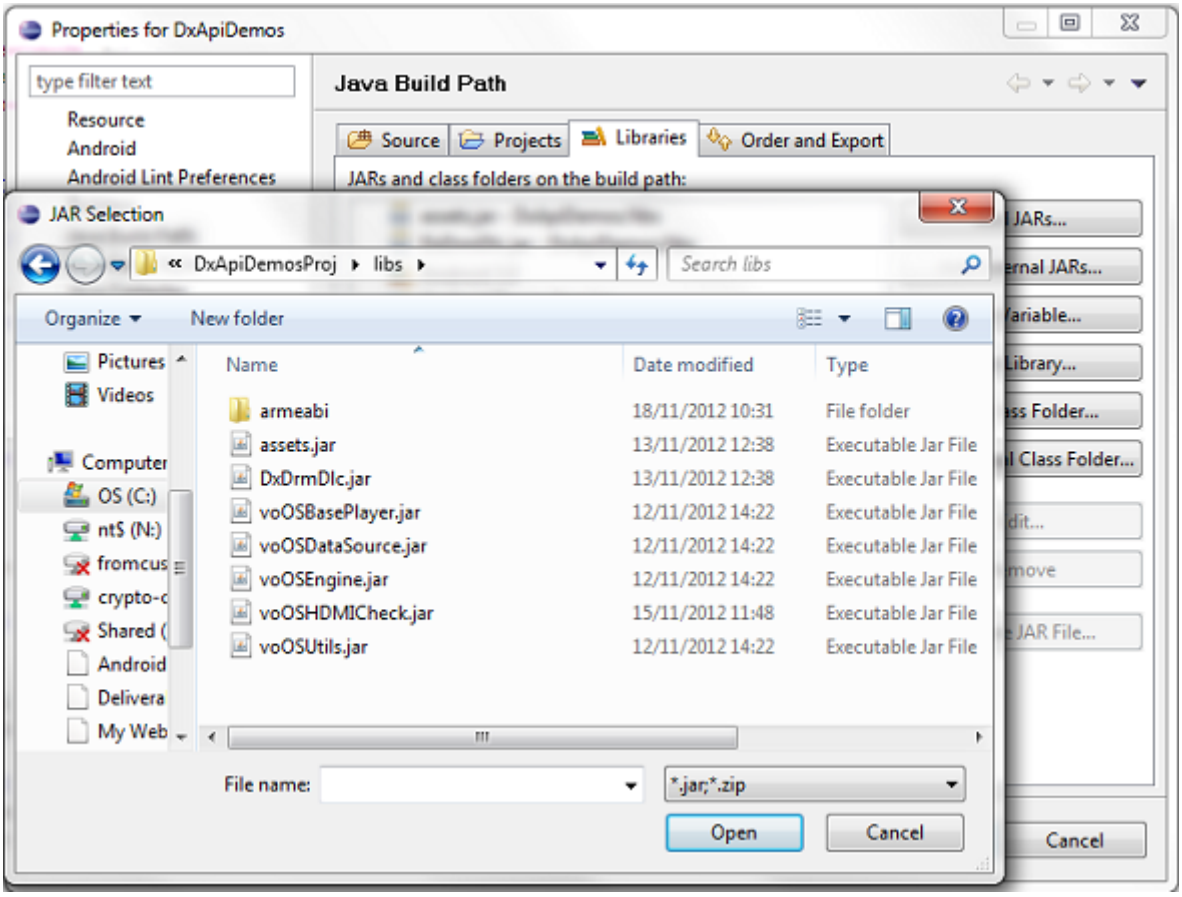
10. Click **Finish**.

Set Project Properties

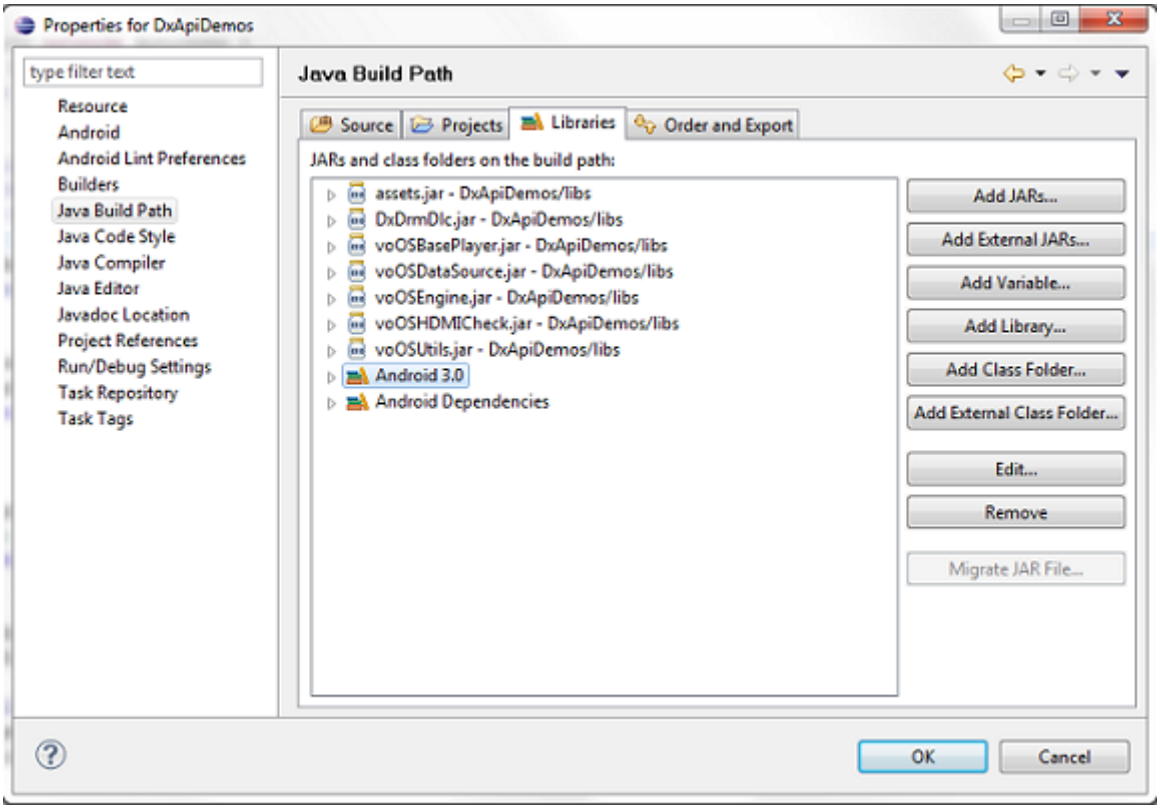
1. Right-click the package root node (project name).
2. In the popup menu select **Properties**.
3. Select **Java Build Path** on the left panel and then the **Libraries** Tab.



4. Click **Add JARs**.
5. Select the `assets.jar`, `DxDrmDlc.jar`, `voOSBasePlayer.jar`, `voOSDataSource.jar`, `voOSEngine.jar`, `voOSHDMICheck.jar` and `voOSUtils.jar` files under <Project name> -> libs


Properties for DxApiDemos
type filter text
Resource
Android
Android Lint Preferences
Java Build Path
Source
Projects
Libraries
Order and Export
JARs and class folders on the build path:
JAR Selection
DxApiDemosProj
libs
Search libs
Organize
New folder
Pictures
Videos
Computer
OS (C:)
Name
Date modified
Type
armeabi 18/11/2012 10:31 File folder
assets.jar 13/11/2012 12:38 Executable Jar File
DxDrmDlc.jar 13/11/2012 12:38 Executable Jar File
voOSBasePlayer.jar 12/11/2012 14:22 Executable Jar File
voOSDataSource.jar 12/11/2012 14:22 Executable Jar File
voOSEngine.jar 12/11/2012 14:22 Executable Jar File
voOSHDMICheck.jar 15/11/2012 11:48 Executable Jar File
voOSUtils.jar 12/11/2012 14:22 Executable Jar File
File name:
.jar;.zip
Open
Cancel
Cancel

6. Click **OK > OK**.


Properties for DxApiDemos
type filter text
Resource
Android
Android Lint Preferences
Builders
Java Build Path
Java Code Style
Java Compiler
Java Editor
Javadoc Location
Project References
Run/Debug Settings
Task Repository
Task Tags
Java Build Path
Source
Projects
Libraries
Order and Export
JARs and class folders on the build path:
assets.jar - DxApiDemos/libs
DxDrmDlc.jar - DxApiDemos/libs
voOSBasePlayer.jar - DxApiDemos/libs
voOSDataSource.jar - DxApiDemos/libs
voOSEngine.jar - DxApiDemos/libs
voOSHDMICheck.jar - DxApiDemos/libs
voOSUtils.jar - DxApiDemos/libs
Android 3.0
Android Dependencies
Add JARs...
Add External JARs...
Add Variable...
Add Library...
Add Class Folder...
Add External Class Folder...
Edit...
Remove
Migrate JAR File...
OK
Cancel

7. Double click `AndroidManifest.xml` in the **Package Explorer**.
8. Add the following user permissions into the manifest, right above the application tag:

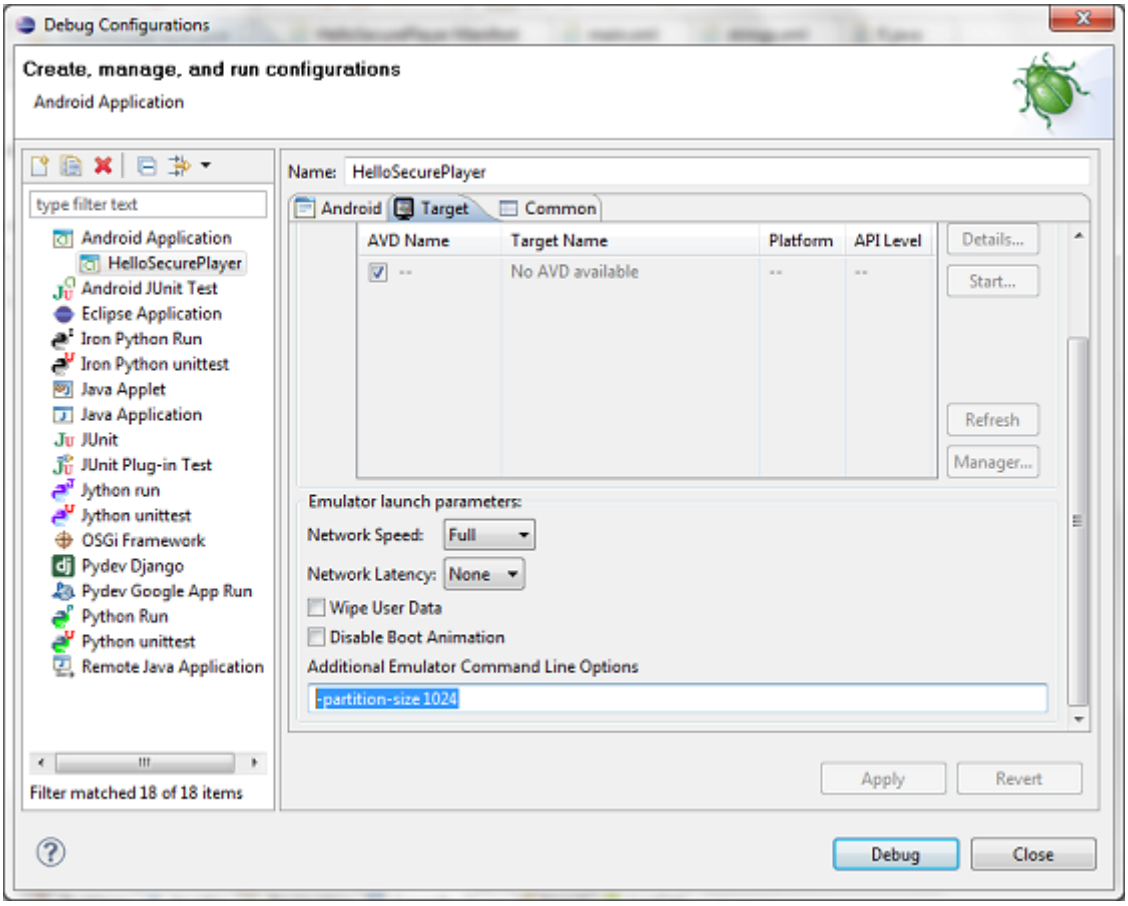
```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<uses-permission android:name="android.permission.DOWNLOAD_WITHOUT_NOTIFICATION"/>
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
```

9. Save the file (Ctrl+S).

Troubleshooting

If you encounter the error `INSTALL_FAILED_INSUFFICIENT_STORAGE` during an attempt to debug the application, proceed as follows:

1. Select in menu: **Run > Debug Configuration**
2. Select Target tab.
3. In the **Additional Emulator Command Line Option** textbox type the following text: `-partition-size 1024`



4. Click **Apply> Close**

Appendix

This appendix contains a glossary and the list of reference documentation.

Glossary

The following table shows typical terms and acronyms found in this document

Term	Definition
IDE	Integrated Development Environment

Reference Documentation

- Connected Sentinel Player SDK Common Integration Guide

Reference number: 21816

- Connected Sentinel Player SDK Android Integration Guide

Reference number: 21697

- Connected Sentinel Player SDK Android API Reference

Reference number: