Getting Started with Android Development – Zebra Android Link-OS[™] SDK – Java – Eclipse



Overview

This Application Note describes the end to end process of designing, packaging, deploying and running an Android application using the Zebra Link-OS[™] Multiplatform SDK and Smartphone Utility.

The sample code used in this Application Note is from the sample code article <u>Connect and</u> <u>Print Over TCP/IP and Bluetooth – Zebra Android Link-OS™ SDK – Java – Eclipse</u>.

Target Audience

The information in this document assumes that you have technical competence with Microsoft Windows, Java development with the Eclipse Integrated Development Environment (IDE), and core programming concepts and rationales.

Feedback

We value your feedback. <u>Please rate the article and then add your comments/suggestions in the text field that appears.</u>

For information on joining the <u>ISV Program</u>, contact a Program Manager in your area.

System Prerequisites

This Android SDK requires the following system prerequisites.

Java Development Kit (JDK)

To develop Android applications, you must install the latest version of the JDK. <u>Download</u> the JDK.

Eclipse IDE for Java Developers

Eclipse is the IDE used to develop Android applications, such as the one featured in this Application Note. <u>Download</u> the IDE.

Note: The version used in this Application Note is Eclipse - Luna Release 2 (4.4.2).

ADT Plugin for Eclipse

The Android Development Tools (ADT) is a plugin for the Eclipse IDE that is designed to give you a powerful, integrated environment in which to build Android applications. <u>Install</u> the ADT plugin.



Android SDK

Google distributes the Android SDK free of charge. It contains sample code, libraries, emulators and tools to help you build applications for the Android platform. See the <u>installation instructions</u>.

Zebra Link-OS Multiplatform SDK

The Zebra Link-OS Multiplatform SDK contains all the required components to develop applications for Zebra label printers.

The SDK includes a Java library (.jar), which provides the means to scan for, connect to, and print on Zebra label printers. <u>Information</u>, including system prerequisites and download links for Zebra Link-OS Multiplatform SDK Zebra Network-Enabled Label Printer is available.

In order to fully test the application you create, you must have a network-enabled Zebra printer capable of understanding ZPL or CPCL.

Bluetooth connectivity is available through the Zebra Link-OS Multiplatform SDK and covered in other available tutorials.

Check out all of the Zebra label printers.

Android Mobile Device

While the Android Emulator, included in the Android SDK, satisfies most of the anticipated requirements for developing Android applications, we recommended that you test all development with a physical device.

Zebra Link-OS Multiplatform SDK supports various printers and mobile devices.

Installations

You must install all of the items listed in the "System Prerequisites" section.

Note: The default installation options for all these prerequisites are satisfactory.



Creating Your Zebra Android Connect and Print Application Project

You will be guided on how to create a mobile application that allows a handheld device to establish a Bluetooth or TCP/IP connection with a Zebra mobile printer.

Note: The sample code used in this Application Note is from the sample code article <u>Connect and Print Over TCP/IP and Bluetooth – Zebra Android Link-OS™ SDK –</u> <u>Java – Eclipse</u>.

To begin, launch Eclipse IDE and locate your desired workspace path.

Note: For the purpose of this application note, we use "C:\MyFirstZebraAndroidApp".

🗢 Workspace Launcher					
Select a workspace					
Eclipse stores your projects in a folder called a workspace. Choose a workspace folder to use for this session.					
Workspace: C:\MyFirstZebraAndroidApp Browse					
Use this as the default and do not ask again OK Cancel					

This section includes instructions on how to:

- Configure the SDK Location
- <u>Create an Android Project</u>
- Design Your Zebra Android Connect and Print Application
- Create Java Helper Classes
- <u>Update ConnectivityDemo.java and Include the Zebra Multiplatform SDK Library for</u> <u>Android</u>
- Run Your Zebra Android Connect and Print Application
- Deploy Your Mobile Application



Configure the SDK Location

- 1. Go to Window -> Preferences.
- 2. Click Android on the left panel.
- 3. Browse to the location of the Android SDK on the right panel.
- 4. Click Apply.

A list of SDK Targets appears.

5. Click **OK** to close the Preferences dialog.

Preferences					
type filter text General Android Ant C/C++ Code Recommen	1. Salas	Android Android Preferences SDK Location: C:\User	2. Browse to SDK locati rs\esuero\AppData\Local\Android\s rgets below is only reloaded once ye	on v and a start of the start o	wse
⊳ Help	1. Selec	me	Vendor	Platform	API
 Help Install/Update Java Library Hover Maven Mylyn Perl EPIC PyDev Remote Systems Run/Debug Target Explorer Team Terminal Validation WindowBuilder XMI 		Android 5.1.1 Google APIs	Android Open Source Project Google Inc.	5.1,1 5.1,1	22 22
			Restore Defa	ults	pply
?			ОК	Car	ncel

Create an Android Project

1. Go to File -> New -> Android Application Project.

Note: Alternatively, you can go to File -> New -> Project, and select Android -> Android Application Project.

- 2. Type AndroidConnectAndPrint as the project name.
- 3. Type com.zebra.isv.connectAndPrint for both Application name and Package name.
- 4. Type ConnectivityDemo next to Create Activity (it must be checked).

Note: You may leave other default inputs unchanged.



New Android Application					
New Android Applicatio	on or most apps begins with an uppercase letter	0			
Application Name:	com.zebra.isv.connectAndPrint				
Project Name:	AndroidConnectAndPrint				
Package Name: 🌢	com.example.androidconnectandprint				
		_			
Minimum Required SDK:0	API 8: Android 2.2 (Froyo)				
Target SDK:0	API 19: Android 4.4 (KitKat)	•			
Compile With:0	API 22: Android 5.1.1 (LOLLIPOP_MR1)	•			
Theme:0	Holo Light with Dark Action Bar	•			
?	< Back Next > Fin	ish Cancel			

- 5. Click Next, leaving the next windows at default, until the Blank Activity window appears.
- 6. In the Blank Activity window, change the activity name to **ConnectivityDemo** and click **Finish** to exit the window.

New Android Application	
Blank Activity Creates a new blank activity with an action bar.	0
(•
Activity Name® ConnectivityDemo	
Layout Name® activity_connectivity_demo	
♀ The name of the activity class to create	
	Cancel



Design Your Zebra Android Connect and Print Application

- 1. Define the variables and values to use in the layout (xml) files later.
 - In the Package Explorer, open strings.xml in the res/values folder.
 - Replace the contents of the "String.xml" file from the "Strings.xml" file from the sample article <u>Connect and Print Over TCP/IP and Bluetooth – Zebra Android Link-</u> <u>OS™ SDK – Java – Eclipse</u>.

For information on joining the <u>ISV Program</u>, contact a Program Manager in your area.

This source code defines the Application Name that appears on the GUI, as well as the color Red for the status bar background when no printer is connected. For more information, go to <u>String Resources</u>.

- Save and close the file.
- 2. Define the layout.
 - In the Package Explorer, expand the res/layout folder. Delete main.xml.
 - In the same folder, create 2 files, "connection_screen_with_status.xml" and "connection_screen.xml".
 - Replace the contents of the two files with the contents of the "connection_screen_with_status.xml" and "connection_screen.xml" files from the sample article <u>Connect and Print Over TCP/IP and Bluetooth – Zebra Android Link-</u> <u>OS™ SDK – Java – Eclipse</u>.

Note: For more information, click <u>Layout Resource</u>.

- Save and close the files.
- 3. Update the AndroidManifest.xml.
 - Replace the contents of the "AndroidManifest.xml" file with the contents of the "AndroidManifest.xml" file from the sample article <u>Connect and Print Over TCP/IP and</u> <u>Bluetooth – Zebra Android Link-OS™ SDK – Java – Eclipse</u>.

Note: The AndroidManifest.xml contains helpful information.

4. Go to res/menu folder to delete the connectivity_demo.xml file settings menu.



Create Java Helper Classes

- 1. Right click on src, hover over New and click Package to create a new Java Package.
- 2. Name the package "com.zebra.isv.util".



3. Create a file "DemoSleeper.java" in the "com.zebra.isv.util" package and replace its contents from the "DemoSleeper.java" file from the sample article <u>Connect and Print Over</u> <u>TCP/IP and Bluetooth – Zebra Android Link-OS™ SDK – Java – Eclipse</u>.

This file contains a static sleep method that ConnectivityDemo.java uses.

4. Create a file "SettingsHelper.java" in the "com.zebra.isv.util" package and replace its contents from the "SettingsHelper.java" file from the sample <u>Connect and Print Over</u> <u>TCP/IP and Bluetooth – Zebra Android Link-OS™ SDK – Java – Eclipse</u>.

This file contains methods used in ConnectivityDemo.java to persist Bluetooth and TCP/IP settings for subsequent recall.

Update ConnectivityDemo.java and Include the Zebra Multiplatform SDK Library for Android

 Replace the source code in ConnectivityDemo.java with the contents of the "ConnectivityDemo.java" file from the sample article <u>Connect and Print Over TCP/IP and</u> <u>Bluetooth – Zebra Android Link-OS™ SDK – Java – Eclipse</u>.

This class is the starting point of the Android application.

- 2. At this point, you should see many errors.
 - **Note:** To resolve the dependencies, right-click the "AndroidConnectAndPrint" project, select **Build Path**, then "Add External Archives...".



🖨 Ja	va - Anc	Iroid	Cor	nectAndPrint/src/com/zebra/is	sv/androidconnectandprint/Co	nneo	tivityDemo.java - Eclipse
File Edit Refactor Source Navigate Search Project Run Window Help							
	• 📫 •		Ţ,	🕽 👜 🗐 📽 New Connection	- 14 14 -	1	· ୬ ⇒ 🗉 🖬 🖳 🔌 📩 🗎 🗹 → 🛛
[🛱 F	Package	Explo	rer	22	□ 🔄 💱 🗸 =	' 🗆	🚺 *ConnectivityDemo.java 🔀
4	Andr 🕑 s 4 🖉	rc		New Go Into Open in New Window	•		1⊖/ 2 * CONFIDENTIAL AND PROPRIET. 3 * 4 * The source code and other 5 * ZIH <u>Corp.</u> and is subject 6 * This source code and any
		>		Open Type Hierarchy Show In	F4 Alt+Shift+W ▶	Ŀ	7 * displayed or distributed, 8 * expressly permitted under
	▷ 📴 g ▷ 🛋 G ▷ 🛋 A	en ioc		Copy Copy Qualified Name Paste	Ctrl+C		9 * 10 * Copyright ZIH <u>Corp</u> . 2010 11 * 12 * ALL RIGHTS RESERVED 13
	⊳ 🏪 b	in 3	C	Delete	Delete	Ŀ	14 package com.zebra.isv.androi
	⊳ 🎦 li ⊳ 🚰 n	bs es		Remove from Context	Ctrl+Alt+Shift+Down	L	16⊖ import android.app.Activity;
	Ā 🔊	inc		Build Path	+	20	Link Source s. Bundle;
	i 💽	:_li ro		Source	Alt+Shift+S ►	₽ °	New Source Folder iew.View;
	P	roj		Refactor	Alt+Shift+T ►	¢#	Use as Source Folder /iew.View.OnC idget.Button
	P appo sadsi	or 👔	1	Import Export			Add External Archives ridget.EditTe Add Libraries ridget.RadioB
			5	Refresh	F5		ridget.RadioG Configure Build Path ridget.TextVi
				Close Project Close Unrelated Projects Assign Working Sets			28 29 import com.zebra.android.com 30 import com.zebra.android.com 31 import com.zebra.android.com
				Profile As Debug As Run As Validate	Þ •		 32 import com.zebra.android.rci 34 import com.zebra.android.pri 35 import com.zebra.android.pri 36 import com.zebra.android.pri 37 import com.zebra.isv.util.De
				Team Compare With	*	Ŀ	38 import com.zebra.isv.util.Se 39 40⊖ /**
				Restore from Local History	,	L	41 * @author MLim3 42 * 43 */
				Android Tools	•	L	44⊝ /** 45 * @author MLim3
				Configure	,	L	45 - 7/ 47⊝ /**
				Properties	Alt+Enter		48 * @author MLim3 49 */
							50 public class ConnectivityDem
							53 private <u>ZepractinterConn</u> 53 private RadioButton btRa 92 54 private <u>ZebraPrinter</u> pri
							55 private TextView statusF 56 private EditText macAddr 57 private Button testButto 58

3. Browse to the location of the "ZSDK_API.jar" file and click **Open**.

The default installation path of this file should be "C:\Program Files\Zebra Technologies\link_os_sdk\android\v2.8.2148\lib\ZSDK_ANDROID_API.jar".

The errors in "ConnectivityDemo.java" should disappear now.

4. Right click on the Project, go to **Java Build Path -> Order and Export** and make sure ZSDK_ANDROID_API.jar is checked.



Properties for AndroidConnect.	AndPrint	
type filter text	Java Build Path	← < ⇒ <
Android Android Lint Preferences Builders Java Build Path Java Code Style Java Compiler Java Editor Javadoc Location Project References	Source Projects Libraries Vorder and Export Build class path order and exported entries: (Exported entries are contributed to dependent projects) Control ConnectAndPrint/src Change Android ConnectAndPrint/gen Android 5.1.1 Android 5.1.1 Android Private Libraries Android Dependencies Androi	Up Down Top
Refactoring History Run/Debug Settings Task Repository Task Tags Validation WikiText		Select All Deselect All
?	0	K Cancel



Run Your Zebra Android Connect and Print Application

- 1. Build and run the application.
 - Right-click the "AndroidConnectAndPrint" project, select Run As -> Android Application.

This runs the application in an Android emulator. The emulator might take a while to start up, depending on the available computer resources and hardware.

Java - AndroidConnectAndPrint/src/com/zebra/isv/connectAndPrint/ConnectivityDemo.java - Eclipse					
File Edit Refactor Source Navigate Search Project Run Window Help					
	🗈 🛓 🖍 New Connection	- 14 14 -	· (♀ 彡 ♥ ■ ■ ■ ■ ≥ ≥ ∎ ■ ■ • 6		
🛱 Package Explor	er 🔀	□ 🔄 😜 🗸 =	- 🖬 🔄 AndroidConnectAndPrint Manifest		
AndroidCo	New Go Into	•	2⊕ * CONFIDENTIAL AND PROPRIETA 14 package com.zebra.isv.connect 15		
	Open in New Window Open Type Hierarchy Show In Copy Copy Qualified Name Paste Delete	F4 Alt+Shift+W ► Ctrl+C Ctrl+V Delete	39 40⊕ /** 41 * @author MLim3 42 * 43 */ 44⊕ /** 45 * @author MLim3 46 */ 47⊕ /** 48 * @author MLim3 49 */ 50 public class ConnectivityDemo		
▲ Ket → → ∞ → ass → bir → bir → bir → 1ib: ▲ 200 res	Remove from Context Build Path Source Refactor	Ctrl+Alt+Shift+Down ► Alt+Shift+S ► Alt+Shift+T ►	os_sd 52 53 54 55 55 55 55 55 55 55 55 55		
	Import Export Refresh Close Project Close Unrelated Projects Assign Working Sets	F5	57 private button testbutton 58 59 /* (non-Javadoc) 60 * @see android.app.Activ 61 */ 62 @Override 63 protected void onCreate(8 64 65 super.onCreate(savedI		
	Profile As Debug As	•	67 68 ipDNSAddress = (EditT		
	Run As	+	1 Android Application		
 An ic_ pro pro pro pro pro pro sappco Sadsac 	Validate Team Compare With Restore from Local History PyDev	۶ ۲	JU 2 Android JUnit Test ex Image: Set		
	Android Tools Configure	*	79 80 testButton = (Button) 81⊖ testButton.setOnClick		
	Properties	Alt+Enter	82 839 public yold on []i		
			2 350 public vold Wheed(ne 2 850 public vold 850 public vold enabl 87 Loope 88 doCon 89 Loope 90 Loope 91 } 92 }).start(); 93 };		

• To run the application, unlock the screen on the emulator by dragging the green lock icon to the right of the screen.





Congratulations! You now have your first Zebra mobile application running on an emulator!

2. To exit the application, click **Back** on the emulator.

To continue development, leave the emulator running as it takes considerable amount of time to start up.

Note: The Bluetooth and TCP/IP connectivity may not be correctly emulated. Test the application on a physical device, which will be covered in the next section.





Deploying Your Mobile Application

After your project is complete, you can export it into an Android Application Package (.APK) file for deployment to an Android Device.

1. Right-click the AndroidConnectAndPrint project, and select "Export...".

The Export dialog appears.

- 2. Expand Android, select Export Android Application and click Next.
- On the Export Android Application, enter the project in this case, it is "AndroidConnectAndPrint" – and click Next.

A dialog appears asking you to enter keystore information.

4. Type the keystore information.

If you do not have this information, type "C:\AndroidKeyStore" (or any other desired folder) as Location, and then type your desired password.

5. Complete the Key Creation form and click **Next**.

Note: You must enter at least the Alias, Password, Confirm, Validity and First and Last Name fields. If you do not, you are not allowed to proceed.

More information on signing your applications is available.

6. On the Destination and key/certificate checks page, type the path for the destination APK file.

Note: The folder that you specify must first exist. If not, create it in Windows Explorer first.

- 7. Click Finish to export the project.
- 8. Copy the .APK file to your device and install your application.

See Also

- For the list of printers and mobile devices that are supported by the Zebra Link-OS Multiplatform SDK, <u>click here</u>.
- For further Zebra Link-OS SDK enabled Android Digital Device sample code and applications, refer to the <u>Zebra Support portal</u>.

For any other information, sample code, white papers, and solutions or to request further content, visit the <u>Zebra Support portal</u>.



Document Control

Version	Date	Description
1.0	Dec 2 2010	Initial Release
1.1	Mar-1 2011	Minor update to See also section
2.0	Jun 28 2012	Updated Chinese link into document
3.0	July 31 2012	Updated Spanish and Portuguese link into document
4.0	July 2015	Updated for most recent Link-OS SDK, eclipse and android SDK

Disclaimer

All links and information provided within this document are correct at time of writing Created for Zebra Global ISV Program by Zebra Development Services.

Note: Google has announced that they will not support Eclipse IDE after December 31 of 2015. We recommend that you migrate to Android Studio.