CTPG Printer SDK Programmer's Manual for Android

Table of Contents

1.0 Overview

The CognitiveSDK for Android is an SDK aimed at development engineers who are developing Android applications for printing on CTPG POS and Label printer. This SDK helps developers to search Cognitive printers via bluetooth, establish connection and printing from android application.

1.1 Features

- Allows search Cognitive printers from android application.
- Allows establish connection from android application.
- Allows printing from android application.
- Allows acquisition printer status from android application.

1.2 CognitiveSDK API

1.2.1 Print API

- Print Setting (Alignment, Line Feed, Text Rotation, Page mode etc)
- Character data Setting (Font, Print position, Double sizing etc)
- Character style setting (Inversion of black and white, Bold, Underline etc)
- Paper feed type setting
- Image printing
- Barcode Printing
- POS and Label Command transmission
- Acquisition response from printer.

1.2.2 Search API

- Search for printer via Bluetooth interface.

1.2.3 Setting API

This API is only for Label Printers and following are the options coming under this

- Print speed setting (High speed, normal, Low speed)
- Feed type setting
- Index setting
- Print type setting

2 Operating Environment

2.1 Android version

- Minimum OS Version 4.0.1
- Maximum OS Version 4.4

2.2 Printers

- A798
- A799
- DLXi
- C Series

2.3 Development Environment

- Android SDK 13 or later
- Java development Kit 6 or later.
- Eclipse Indigo
- ADT Plugin for Eclipse

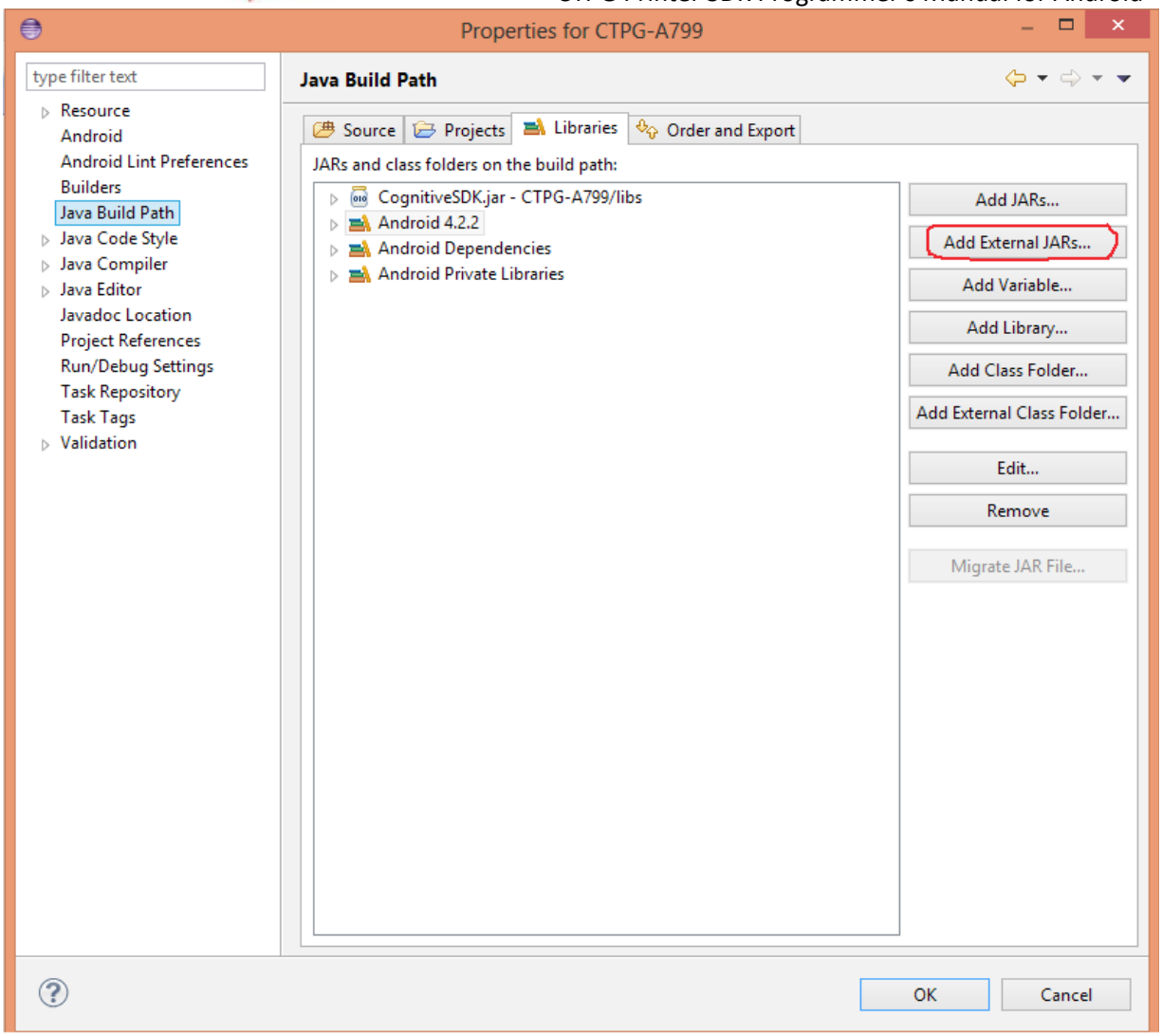
3 Programming Guide

This section explains how to write programs in the application development using CognitiveSDK.

3.1 How to include CognitiveSDK in the android application.

Following are the steps to include CognitiveSDK in android application.

- Create a new project in Eclipse
- Copy provided jar file (CognitiveSDK.jar) into libs folder
- In the libraries tab of the target project properties confirm that the JAR file added is registered in the build path. If it has not been added then , add the JAR file using "Add External JARs"



- Select your project from "Package Explorer" , right click on it press "Refresh".
- Write the package import declaration in the *.java source file(s) of the application you would like to use this SDK in as follows:

```
import com.cognitive.printer.io.POSPrinterIO;
import com.cognitive.printer.io.POSPrinterIO.PrintMode;
import com.cognitive.util.BitmapConvertor;
```

3.2 CognitiveSDK Programming Guide

In this section explains how to use different API of this SDK in our android application.

3.2.1 Search Printers

To search a CTPG printer user searchPrinters() method of ConnectionManager API. In this method developers need to pass device type parameter. Following is the sample code for searching a Printer.

```
list=ConnectionManager.searchPrinters(DevType.BLUETOOTH);

// Where list is a Arraylist.
```

3.2.2 Establish Connection

To create connection between android device and CTPG printers , developer need to use following piece of code :-

```
ConnectionManager connection;// ConnectionManager Object

connection=ConnectionManager.getConnection(DevType.BLUETOOTH);
connection.setConnectionListener(context);
connection.openConnection(address);//Mac Address of the printer
printer=(PoSPrinter) PrinterFactory.getPrinter(PRINTER_MODEL)

// Use Typecast "PoSPrinter" when Printer is POS, Use Typecast "LabelPrinter"
if printer is Label Printer.

printer.setConnection(connection);

When want to disconnect use closeConnection() method.

connection.closeConnection();
```

3.2.3 Text Printing

To print a text using CognitiveSDK developer need to do following things : -

- Initialize depending upon the type of Printers like below

```
POSPrinterIO buffer=new POSPrinterIO(); // Creating Printer
Buffer
buffer.addInitializePrinter();// Initializing
```

- Specify text properties like Alignment, italics, upside-down etc.

```
buffer.addTextItalic(TRUE/FALSE);
buffer.addTextUpsideDown(TRUE/FALSE);
buffer.addTextInvertColor(TRUE/FALSE);
buffer.addTextRotation(TRUE/FALSE);
```

- Convert printed text into byte data and add it to buffer as following

```
byte[] data=DATA_TO_PRINT.getBytes();
buffer.addTextData(data);
```

- Use sendCommand() method to print the text

```
printer.sendCommand(buffer);
```

For **Label Printer** developer need to use following pieces of code : -

```
LabelPrinterIO label=new LabelPrinterIO();
label.addHeader(Mode.ASCII, 0, 200, 150, 1);//Add Header
label.addPitch(Pitch.DPI_100);//Add Pitch value
label.addText(FONT, SPACING, ROTATION, X-MULTI ,Y-MULTI, X axis Value, Y axis
      value, TEXT);
label.addEnd();
printer.sendCommand(label);
```

3.2.4 Image Printing

Following are the steps to print an image using CognitiveSDK for POS Printer.

- Get the image into main memory , convert it into monochrome bitmap, convert it into byte data

```
Bitmap bmp=BitmapFactory.decodeStream(PATH OF IMAGE);
BitmapConvertor convert=new BitmapConvertor();
byte[] image=convert.getMonochromeBitmap(bmp);
```

- Create buffer depending upon the printer and reset it using addResetPrinter() method.

```
POSPrinterIO buffer=new POSPrinterIO();
buffer.addResetPrinter();
```

- Add alignment type and byte image data to printer's buffer and print it.

```
buffer.addAlignment(ALIGNMENT);
buffer.addLogo(image);
printer.sendCommand(buffer);
```

For **Label Printer** developer should use following code : -

```
LabelPrinterIO label=new LabelPrinterIO();
label.addHeader(Mode.ASCII, 0, 100, 600, 0);//Add Header
label.addWidth(400); //Add print width use 200 for 2 inch Printer
label.addGraphic(GraphicType, X axis value, Y axis Value, IMAGE);
label.addEnd();
printer.sendCommand(label);
```

3.2.5 Barcode Printing

Developers need to addBarcode() method of POSPrinterIO and LabelPrinterIO class to print Barcode in POS and Label printer respectively. Following are the sample for both the printers.

- **POS Printer**

```
POSPrinterIO buffer=new POSPrinterIO();
buffer.addResetPrinter();
buffer.addAlignment(ALIGNMENT);
buffer.addBarcode(HEIGHT, WIDTH, HRI_PLACE, BARCODE_TYPE, DATA);
printer.sendCommand(buffer);
```

- **Label Printer**

```
LabelPrinterIO label=new LabelPrinterIO();
label.addHeader(Mode.ASCII, 0, 200, 150, 1);//Add Header
label.addPitch(Pitch.DPI_100);// Add Pitch value
label.addBarcode(BARCODE_TYPE, MODIFIER, X, Y , HEIGHT, TEXT);
label.addEnd();
printer.sendCommand(label);
```

3.2.6 QR Code Printing

In this SDK QR Code printing is only available for POS Printer and it can be use as following :-

```
POSPrinterIO buffer=new POSPrinterIO();
buffer.addResetPrinter();
buffer.addAlignment(ALIGNMENT);
buffer.addQRCode(MODEL, SIZE, OPTION, DATA);
printer.sendCommand(buffer);
```

3.2.7 Raw IO Command

CognitiveSDK provides writeData() method where developer can send Hexadecimal command and can receive response from printer. Following is the sample to send hexadecimal command to the printer.

```
byte[] cmd=hexStringToByteArray(COMMAND as STRING);//convert String to
                                                    hexadecimal command.
byte[] params=getParams();
connection.writeData(COMMAND, INITIAL_POSITION, MAX_LENGTH);
```

Receive Response from Printer

```
byte[] read=new byte[100];
int size=connection.readData(read, 0, read.length);//connection is the object
                                                    of ConnectionManager class
String str=new String(read, 0, size)
TextView status = new TextView(this);
status.setText("Printer Response: "+str);
```

3.2.8 Most commonly methods (POS)

- Printer Status

```
PrinterStatus ps = printer.getStatus();

if (ps.isBusy()) {
showToast("Busy State");
}
if (ps.isCoverOpen()) {
showToast("Cover Open");
}
if (ps.isDrawerOpen()) {
showToast("Drawer Open");
}
if (ps.isPaperLow()) {
showToast("Paper Low");
}if (!ps.isOnline())
showToast("Printer Offline");
}
```

- Generate Beep

```
POSPrinterIO buffer=new POSPrinterIO();
buffer.addResetPrinter();
buffer.addTone();
printer.sendCommand(buffer);
```

- Test Print

```
POSPrinterIO buffer=new POSPrinterIO();
buffer.addResetPrinter();
buffer.addTestForm();
printer.sendCommand(buffer);
```

- Partial Cut

```
POSPrinterIO buffer=new POSPrinterIO();
buffer.addResetPrinter();
buffer.addFeedLines(5);// Number of line feed
buffer.addCut(CutType.PARTIAL_CUT);
printer.sendCommand(buffer);
```

- Full Cut

```
POSPrinterIO buffer=new POSPrinterIO();
buffer.addResetPrinter();
buffer.addFeedLines(5);
```

```
buffer.addCut(CutType.FULL_CUT);
printer.sendCommand(buffer);
```

3.2.9 Most commonly methods (Label)

- Printer Status

```
StatusMessage message=printer.getPrinterStatus();
showToast(message.getCode());
showToast(message.getStatusMessage());
```

- Generate Beep

```
LabelPrinterIO buffer=new LabelPrinterIO();
buffer.addHeader(Mode.ASCII, 0, 0, 0, 0);
buffer.addBeep(2);
buffer.addEnd();
printer.sendCommand(buffer);
```

- Test Print

```
LabelPrinterIO buffer=new LabelPrinterIO();
buffer.addPrintTestLabel();
printer.sendCommand(buffer);
```

- Darkness Setting

```
LabelPrinterIO label=new LabelPrinterIO();
label.addHeader(Mode.ASCII, 0, 0, 0, 0);
label.setDarkness(VALUE);
label.setWrite();
label.addEnd();
printer.sendCommand(label);
```

- Width Setting

```
LabelPrinterIO label=new LabelPrinterIO();
label.addHeader(Mode.ASCII, 0, 0, 0, 0);
label.setWidth(WIDTH_VALUE)
label.setWrite();
label.addEnd();
printer.sendCommand(label);
```

- Print Mode Settings

```
LabelPrinterIO label=new LabelPrinterIO();
label.addHeader(Mode.ASCII, 0, 0, 0, 0);
```

```
label.setPrintMode(PRINT_MODE) // DT, TT or AUTO
label.setWrite();
label.addEnd();
printer.sendCommand(label);
```

- Feed Type Settings

```
LabelPrinterIO label=new LabelPrinterIO();
label.addHeader(Mode.ASCII, 0, 0, 0, 0);
label.setPrintMode(FEED_TYPE) // GAP, NOTCH, BAR
label.setWrite();
label.addEnd();
printer.sendCommand(label);
```