USER MANUAL

ANDROID SDK FOR P3L

CUSTOM®

CUSTOM S.p.A.
Via Berettine 2/B
43010 Fontevivo (PARMA) - Italy
Tel. : +39 0521-680111
Fax : +39 0521-610701
http: www.custom.biz

Customer Service Department:
www.custom4u.it

© 2023 CUSTOM S.p.A. – Italy.
All rights reserved. Total or partial reproduction of this manual in whatever form, whether by printed or electronic means, is forbidden. While guaranteeing that the information contained in it has been carefully checked, CUSTOM S.p.A. and other entities utilized in the realization of this manual bear no responsibility for how the manual is used. Information regarding any errors found in it or suggestions on how it could be improved are appreciated. Since products are subject to continuous check and improvement, CUSTOM S.p.A. reserves the right to make changes in information contained in this manual without prior notification.

The pre-installed multimedia contents are protected from Copyright CUSTOM S.p.A. Other company and product names mentioned herein may be trademarks of their respective companies. Mention of third-party products is for informational purposes only and constitutes neither an endorsement nor a recommendation. CUSTOM S.p.A. assumes no responsibility with regard to the performance or use of these products.

TABLE OF CONTENTS

INTRODUCTION

CONNECT THE PRINTER

Initialize PosManager 8

Connection action 9

Connecting to printers through BluetoothEdrConfigBean 10

WiFi Connection 11

USB Connection 12

Connecting the printer through the device 15

Com Connection 16

Connecting Printers through the device 18

Disconnect 19

PRINT MANAGEMENT

Self-test page test 20

Text Printing 21

Barcode Printing 22

Picture Printing 23

Template Printing 24

JAR SEVERAL COMMONLY USED CLASS API DETAILS

PosManager 25
CommonSetting 27
TextSetting 29
BarcodeSetting 34
BitmapSetting 37
EscCmd 38

INTRODUCTION

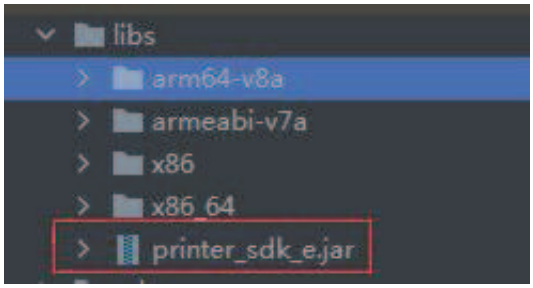
printer_sdk is a jar library for controlling printers under the Android platform, which facilitates the interaction between APP and printer information.

1.1 Integration step

By integrating the Android SDK into the application, the default developer is already familiar with the basic use of the IDE (Eclipse or Android Studio) and has a certain knowledge base of Android programming. Development environment: Java 8, Android Studio 3.2 or above;

1. Go to app > libs, if there is no libs, create a new one; If the project is exported from eclipse, the libs directory is directly in the root directory, remember this;

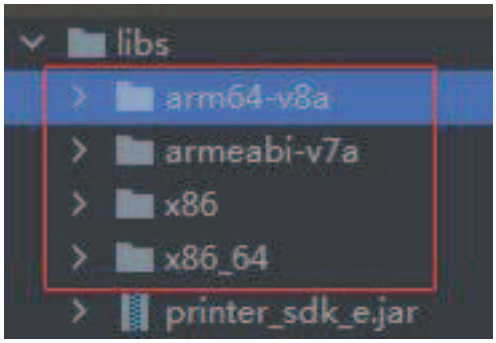
2. Copy the jar package of printer_sdk_e.jar to the labs package, as shown below:



3. Find build.gradle(Module:app), then find dependencies and add :implementation files('libs/printer_sdk_e.jar') as shown below:

```
dependencies {
    implementation fileTree(include: ['*.jar'], dir: 'libs')
    androidTestImplementation('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    implementation 'com.android.support:appcompat-v7:+'
    implementation 'com.android.support.constraint:constraint-layout:1.0.2'
    testImplementation 'junit:junit:4.12'
    implementation files('libs/printer_sdk_e.jar')
```

4. So library required to add com connection




```
sourceSets {
    main {
        jniLibs.srcDirs = ['libs']
    }
}
```

CONNECT THE PRINTER

Printer connection methods: Bluetooth, WiFi, USB, serial port, etc. Connection steps are as follows:

Initialize PosManager

- posStatusCallback Callback for printing connection status. If status is 1, the printer connection is disconnected from 0.
- posMsgCallback Indicates the callback interface for printer information. s indicates the callback message

```
PosManager posManager;
private void initPrinter()
        {
        posManager=new PosManager(this)
                {
                @Override
                public void printerConnectCallback(int status, String msg
                        {
                        updateUI(status,msg);
                        }
                @Override
                public void printerMsgCallback(int status, String msg)
                        {
                        updatePrinterMsg(status,msg);
                        }
                };
        BaseApplication.getInstance().setPosManager(posManager);
        }
```

Connection action

- Bluetooth Connection

- Obtaining the Bluetooth device

```
private void showBluetoothDeviceChooseDialog()
        {
        BluetoothDeviceChooseDialog bluetoothDeviceChooseDialog = new BluetoothDeviceChooseDialog();
        bluetoothDeviceChooseDialog.setOnDeviceItemClickListener(new
        BluetoothDeviceChooseDialog.onDeviceItemClickListener()
                {
                @Override
                public void onDeviceItemClick(BluetoothDevice device)
                        {
                        BluetoothEdrConfigBean configObj = new BluetoothEdrConfigBean(device);
                        isConfigPrintEnable(configObj);
                        }
                });
        BluetoothDeviceChooseDialog.show(MainActivity.this.getFragmentManager(), null);
        }
```

Connecting to printers through BluetoothEdrConfigBean

```
BluetoothEdrConfigBean bluetoothEdrConfigBean = (BluetoothEdrConfigBean) configObj;
        posManager.connectDevice(bluetoothEdrConfigBean);
```

WiFi Connection

- Obtaining the WiFi device:

```
String ip = “192.168.1.87”;
int port = 9100;
WiFiConfigBean configObj = new WiFiConfigBean(ip, port);
```

- Connect the printer through the Wi-Fi WiFiConfigBean

```
WiFiConfigBean wiFiConfigBean = (WiFiConfigBean) configObj;
        posManager.connectDevice(wiFiConfigBean);
```

USB Connection

- Obtaining the USB device

```
public class UsbDeviceChooseDialog extends DialogFragment
        {
        private final String TAG = getClass().getSimpleName()+"ComPrinter";
        private Context mContext;
        private ListView lvContent;
        private TextView tvEmpty;
        private AdapterView.OnItemClickListener mListener;
        private List<UsbDevice> mList;
        private UsbDeviceAdapter mAdapter;
        private UsbManager mUsbManager;
        private UsbDeviceReceiver mUsbReceiver;

        @Override
        public void onAttach(Activity activity)
                {
                super.onAttach(activity);
                mContext = activity;
                }
        @NonNull
        @Override
        public Dialog onCreateDialog(Bundle savedInstanceState)
                {
                LayoutInflater inflater = getActivity().getLayoutInflater();
                View view = inflater.inflate(R.layout.dialog_choose_usb_device, null);
                initView(view);
                setListener();
                initData();
                setAdapter();
                registerUsbReceiver();
                AlertDialog.Builder builder = new AlertDialog.Builder(mContext);
                builder.setView(view).setCancelable(true).setNegativeButton(R.string.dialog_cancel, null);
                return builder.create();
                }
                private void registerUsbReceiver()
                        {
                        Log.i(TAG, "registerUsbReceiver: ");
                        mUsbReceiver = new UsbDeviceReceiver(new UsbDeviceReceiver.CallBack()
                                {
                                @Override
                                public void onPermissionGranted(UsbDevice usbDevice)
                                        {
                                        LogUtils.d(TAG,"onPermissionGranted:"+usbDevice.getDeviceId());
                                        }
                                @Override
                                public void onDeviceAttached(UsbDevice usbDevice)
                                        {
                                        LogUtils.d(TAG,"onDeviceAttached:"+usbDevice.getDeviceId());
                                        mList.add(usbDevice);
                                        mAdapter.notifyDataSetChanged();
                                        tvEmpty.setVisibility(View.GONE);
                                        }
                                @Override
                                public void onDeviceDetached(UsbDevice usbDevice)
                                        {
                                        LogUtils.d(TAG,"onDeviceDetached:"+usbDevice.getDeviceId());
                                        mList.remove(usbDevice);
```

```
                        mAdapter.notifyDataSetChanged();
                        if (mList.size() == 0)
                                {
                                tvEmpty.setVisibility(View.VISIBLE);
                                }
                        }
                });
                IntentFilter intentFilter = new IntentFilter();
                intentFilter.addAction(UsbManager.ACTION_USB_DEVICE_ATTACHED);
                intentFilter.addAction(UsbManager.ACTION_USB_DEVICE_DETACHED);
                mContext.registerReceiver(mUsbReceiver, intentFilter);
        }
        private void initView(View view)
                {
                lvContent = (ListView) view.findViewById(R.id.lv_dialog_choose_usb_device);
                tvEmpty = (TextView) view.findViewById(R.id.tv_dialog_choose_usb_device_empty);
                }
        private void setListener()
                {
                lvContent.setOnItemClickListener(mListener);
                }
        private void initData()
                {
                mList = new ArrayList<>();
                mUsbManager = (UsbManager) mContext.getSystemService(Context.USB_SERVICE);
                HashMap<String, UsbDevice> deviceList = mUsbManager.getDeviceList();
                LogUtils.d(TAG, “deviceList size = “ + deviceList.size());
                Iterator<UsbDevice> deviceIterator = deviceList.values().iterator();
                while (deviceIterator.hasNext())
                        {
                        UsbDevice device = deviceIterator.next();
                        LogUtils.d(TAG, “device getDeviceName” + device.getDeviceName());
                        LogUtils.d(TAG, “device getVendorId” + device.getVendorId());
                        LogUtils.d(TAG, “device getProductId” + device.getProductId());
                        mList.add(device);
                        }
                if (mList.size() == 0
                        {
                        tvEmpty.setVisibility(View.VISIBLE);
                        }
                }
        private void setAdapter()
                {
                mAdapter = new UsbDeviceAdapter(mContext, mList);
                lvContent.setAdapter(mAdapter);
                }
                public void setOnItemClickListener(AdapterView.OnItemClickListener listener)
                {
                mListener = listener;
                }
        @Override
        public void onDismiss(DialogInterface dialog)
                {
                super.onDismiss(dialog);
                LogUtils.v(TAG, “onDismiss”);
                mContext.unregisterReceiver(mUsbReceiver);
                }
}
private void showUSBDeviceChooseDialog()
        {
        final UsbDeviceChooseDialog usbDeviceChooseDialog = new UsbDeviceChooseDialog();
```

```
        usbDeviceChooseDialog.setOnItemClickListener(new AdapterView.OnItemClickListener
                (
                        {
                        @Override
                        public void onItemClick(AdapterView<?> parent, View view, int position, long id)
                                {
                                UsbDevice mUsbDevice = (UsbDevice) parent.getAdapter().
                                getItem(position);
                                PendingIntent mPermissionIntent;
                                if(Build.VERSION.SDK_INT>=31) {//zac add android 12 20230403
                                mPermissionIntent = PendingIntent.getBroadcast(MainActivity.this, 0,
                                new Intent(MainActivity.this.getApplicationInfo().packageName),
                                PendingIntent.FLAG_IMMUTABLE);
                                }
                                else {
                                mPermissionIntent = PendingIntent.getBroadcast(MainActivity.this, 0,
                                new Intent(MainActivity.this.getApplicationInfo().packageName), 0);
                                }
                                   configObj = new UsbConfigBean(BaseApplication.getInstance(), mUsb-
                                   Device,
                                mPermissionIntent);
                                isConfigPrintEnable(configObj);
                                usbDeviceChooseDialog.dismiss();
                        }
        }
                );
                usbDeviceChooseDialog.show(getFragmentManager(), null);
}
```

Connecting the printer through the device

```
UsbConfigBean usbConfigBean = (UsbConfigBean) configObj;
posManager.connectDevice(usbConfigBean);
```

Com Connection

- Obtaining the com device

```
private void initDevice()
        {
        SerialPortFinder serialPortFinder = new SerialPortFinder();
        // device
        mDevices = serialPortFinder.getAllDevicesPath();
        if (mDevices.length == 0)
                {
                mDevices = new String[]
                        {
                        getString(R.string.str_no_devices)
                        };
                }
        // Baud rate
        mBaudrates = getResources().getStringArray(R.array.baudrates);
        mDeviceIndex = mDevices.length - 1;
        mDeviceIndex = mDeviceIndex >= mDevices.length ? mDevices.length - 1 : mDeviceIndex;
        mBaudrateIndex = 0;
        }
private void initSpinners()
        {
        ArrayAdapter<String> deviceAdapter =
        new ArrayAdapter<String>(this,R.layout.spinner_default_item,mDevices);
        deviceAdapter.setDropDownViewResource(R.layout.spinner_item);
        mSpComPath.setAdapter(deviceAdapter);
        mSpComPath.setOnItemSelectedListener(this);
        ArrayAdapter<String> baudrateAdapter =
        new ArrayAdapter<String>(this, R.layout.spinner_default_item, mBaudrates);
        baudrateAdapter.setDropDownViewResource(R.layout.spinner_item);
        mSpBps.setAdapter(baudrateAdapter);
        mSpBps.setOnItemSelectedListener(this);
        mSpComPath.setSelection(mDeviceIndex);
        mSpBps.setSelection(mBaudrateIndex);
        }
/**
* com
*/
private void showCOMDeviceChooseDialog()
        {
        View comSelectView = View.inflate(this,R.layout.dialog_com_select,null);
        TextView title = comSelectView.findViewById(R.id.tv_dialog_title);
        title.setText("Title");
        mSpComPath = comSelectView.findViewById(R.id.sp_com_path);
        mSpBps = comSelectView.findViewById(R.id.sp_bps);
        initDevice();
        initSpinners();
        Button btn_cancle = comSelectView.findViewById(R.id.negativeButton);
        Button btn_comfirm = comSelectView.findViewById(R.id.positiveButton);
        final AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setView(comSelectView);
        final AlertDialog dialog = builder.create();
        dialog.setCanceledOnTouchOutside(false);
        dialog.show();
        btn_cancle.setOnClickListener(new View.OnClickListener()
                {
                @Override
                public void onClick(View view)
```

```
                {
                dialog.dismiss();
                }
        });
btn_comfirm.setOnClickListener(new View.OnClickListener()
        {
        @Override
        public void onClick(View view)
                {
                tv_device_selected.setText(mDevices[mDeviceIndex]+" - "+mBaudrates[mBaudrateIndex]);
                tv_device_selected.setTag(BaseEnum.HAS_DEVICE);
                dialog.dismiss();
                }
        });
}
@Override
public void onItemSelected(AdapterView<?> adapterView, View view, int i, long l)
        {
        switch (adapterView.getId())
                {
                case R.id.sp_com_path:
                mDeviceIndex = i;
                mSerialPortConfigBean.setFile(new File(mDevices[mDeviceIndex]));
                break;
                case R.id.sp_bps:
                mBaudrateIndex = i;
                mSerialPortConfigBean.setBaudrate(Integer.parseInt(mBaudrates[mBaudrateIndex]));
                break;
                }
        }
@Override
public void onNothingSelected(AdapterView<?> adapterView)
}
```

Connecting Printers through the device

posManager.connectDevice(mSerialPortConfigBean);

Disconnect

```
private void doDisConnect()
        {
        if (posManager != null)
                {
                posManager.disconnectPrinter();
                }
        setPrintEnable(false);
        }
```

PRINT MANAGEMENT

Make sure the printer is connected before printing.

Self-test page test

```
private void selfTestPrint()
        {
        posManager.writeAsync(new byte[]{0x12,0x54});
        }
```

Text Printing

```
private String printStr;
private String mChartsetName = “UTF-8”;
private ESCFontTypeEnum curESCFontType = null;
private boolean IsEscSmallCharactor=false;
private boolean IsAntiWhite=false;
private boolean IsDoubleWidth=false;
private boolean IsDoubleHeight=false;
private boolean IsBold=false;
private boolean IsUnderline=false;
private int txtAlign=0;
private void escPrint()
        {
        posManager.writeAsync(new EscCmdManager()
        .setTextAlign(txtAlign)
        .setTextChartSet(mChartsetName)//ChartSet:UTF-8
        .setTextBold(IsBold)//Bold
        .setTextFont(curESCFontType)//add text Font
        .setTextUnderLine(IsUnderline)
        .setTextAntiWhite(IsAntiWhite)
        .setTextFontDoubleWidthHeight(IsDoubleWidth, IsDoubleHeight)
        .setTextSmallFont(IsEscSmallCharactor)
        .setTxtLineSpacing(getInputLineSpacing())
        .setTxtPrintPosition(10, 10)
        .addText(printStr)
        .addLFCRCmd()
        .addLineFeed()
        .build());
        }
```

Barcode Printing

```
private BarcodeType barcodeType=BarcodeType.CODE128;
private String barcodeContent=”code128”;
private void escPrint()
        {
        posManager.writeAsync(new EscCmdManager()
        .setAlignCenter()
        .setBcWideAndHeight(3,72)//bar width height
        .setBcTextPosition(BarcodeStringPosition.BELOW_BARCODE)
        .addBarcode(barcodeType,barcodeContent)//add Barcode
        .addMutiLineFeeds(5)
        .build());
        }
```

Picture Printing

```
private int bmpPrintWidth = 48;
private Bitmap mBitmap = AssetsUtils.getBmFromAssetsFile(this, “tmp01.jpg”);
private void escPrint()
        {
        posManager.writeAsync(new EscCmdManager()
        .setBmpLimitWidth(bmpPrintWidth)
        .setAlignCenter()
        .setBmpCmdMode(BmpPrintMode.MODE_SINGLE_FAST)
        .addBmp(mBitmap)
        .addMutiLineFeeds(5)
        .addAllCutter()
        .build());
        }
```

Template Printing

```
private void testTemplate()
        {
        String strPrice = “10.0”;
        String strTotal = “20.0”;
        String strweight = “30.0”;
        StringBuilder sb = new StringBuilder(“”);
        sb.append(“Print Ticket\n\r”);
        sb.append(String.format(“Weight:%s kg\n\r”, strweight));
        sb.append(String.format(“Price:%.2f Yuan/kg\n\r”, Float.parseFloat(strPrice)));
        sb.append(String.format(“Total:%.2f Yuan\n\r”, Float.parseFloat(strTotal)));
        posManager.writeAsync(new EscCmdManager()
        .setTextAlign(2)
        .setTextChartSet(“GBK”)
        .addText(sb.toString())
        .addLineFeed()
        .setBmpLimitWidth(48)
        .setAlignCenter()
        .setBmpCmdMode(BmpPrintMode.MODE_SINGLE_FAST)
        .addBmp(bmFromAssetsFile01)
        //Command mode print bar use command to print barcode
        .setAlignLeft()
        .setBcWideAndHeight(3,72)
        .setBcTextPosition(BarcodeStringPosition.BELOW_BARCODE)
        .addBarcode(BarcodeType.CODE128,”9876543210”)
        .addMutiLineFeeds(5)
        .addAllCutter()
        .build());
        }
```

JAR SEVERAL COMMONLY USED CLASS API DETAILS

PosManager

Printer connection, disconnect, print management control class

- public boolean isPrinterConncect()
  get value of printer connect status
  Returns:
  Printer connected

- public void write(byte[] bytes)
  write data
  Parameters:
  bytes -

- public void writeAsync(byte[] bytes)
  write data
  Parameters:
  bytes -

- public void connectDevice(Object device)
  Connect the printer
  Parameters:
  device -BluetoothEdrConfigBean
  -WiFiConfigBean

- public void disconnectPrinter()
  disconnect printer

- public abstract void posStatusCallback(PrinterInterface posInterface, int status)
  get connection status
  Parameters:
  posInterface -
  status - 1 Connect
  - 0 disconnect

- public abstract void posMsgCallback(PrinterInterface posInterface, String msg)
  get msg from printer/scale
  Parameters:
  posInterface -
  msg -

- public void getPrintStauts(int cmdType)
  get status

- public void getPrintStauts()
  get status

- public void sendFileToPrinter(String path, int mIntervalTimes, int mRepeat)
  Bin File test
  Path-bin file path
  mIntervalTimes -Interval time
  mRepeat-Print times

- public void cancelSendFile()
  Cancel bin printing

CommonSetting

- public void setAbsolutionPositionN(int absolutionPositionN)
  - Suitable for Esc, pin printing
  Absolute fixed point positioning
  Based on the current left limit, shift the print head right (n1+n2×256) / 60 inches. If the distance exceeds the right-hand limit, this setting is invalid. The maximum setting distance is 10.6 inches.
  Parameters:
  absolutionPositionN - n ≤ 636; n1= n % 256; n2= n/256;

- public void setPageWidth(int pageWidth)
  For Cpcl, ESC
  page width setting
  Parameters:
  pageWidth - CPCL:In-Dot ESC IN-mm: n

- public LableSizeBean getLableSizeBean()
  Get label size object
  Returns:

- public void setLableSizeBean(LableSizeBean lableSizeBean)
  Label size setting
  -for Tsc, Zpl
  -For Tsc, Zpl
  Parameters:
  lableSizeBean -
  getLabelGap
  public int getLabelGap()
  Get Label Gap
  Returns:

- public void setAlign(int align)
  For Pin, Esc
  Esc(ALIGN_LEFT, ALIGN_MIDDLE, ALIGN_RIGHT)]
  align setting [Esc only support ALIGN_LEFT, ALIGN_MIDDLE, ALIGN_RIGHT]
  Parameters:
  align - ALIGN_LEFT, ALIGN_MIDDLE, ALIGN_RIGHT, ALIGN_BOTH_SIDES
  [Esc only support ALIGN_LEFT, ALIGN_MIDDLE, ALIGN_RIGHT]

- public int getEscLineSpacing()

- public void setEscLineSpacing(int n)
  For ESC
  EPOS Line space setting, n * 0.125 mm
  Parameters:
  n - n * 0.125 mm , range: 0<= n <= 255

- public int getEscCharSpacing()

- public void setEscCharSpacing(int n)

- public byte[] setPointXY(int x, int y)

- public byte[] setBlackLabelPlus(int plus)

- public byte[] setBlackLabelSub(int sub)

- public int getxSimLineSpacing()

- public void setxSimLineSpacing(int spacing)

This command sets the line spacing between successive printed text lines (dot row). Each dot row is 0.125mm. 0 ≤ n ≤ 40) decimal= . If n value is greater than 0x28 or Decimal 40, then n will be set to 0x28)Decimal 40).

Parameters:

spacing - 0-40

TextSetting

- public SettingEnum getBold()

- public void setBold(SettingEnum setting)
  For ESC, Pin
  Bold Setting D
  Parameters:
  setting - Enable, Disable

- public SettingEnum getItalic()

- public void setItalic(SettingEnum setting)
  italic setting
  Parameters:
  setting - Enable, Disable

- public SettingEnum getUnderline()

- public void setUnderline(SettingEnum underline)
  For ESC, Pin
  underline setting
  Parameters:
  underline - Enable, Disable

- public SettingEnum getDoublePrinting()

- public void setDoublePrinting(SettingEnum doublePrinting)
  For Pin
  double printint setting
  Parameters:
  doublePrinting - Enable, Disable

- public int getAlign()

- public void setAlign(int align)
  For Pin, ESC
  ESC(ALIGN_LEFT, ALIGN_MIDDLE, ALIGN_RIGHT)]
  align setting [ESC only support ALIGN_LEFT, ALIGN_MIDDLE, ALIGN_RIGHT]
  Parameters:
  align - ALIGN_LEFT, ALIGN_MIDDLE, ALIGN_RIGHT, ALIGN_BOTH_SIDES
  [ESC only support ALIGN_LEFT, ALIGN_MIDDLE, ALIGN_RIGHT]

- public SettingEnum getDoubleWidth()

- public void setDoubleWidth(SettingEnum doubleWidth)
  For ESC, Pin
  double width setting
  Parameters:
  doubleWidth - Enable, Disable

- public SettingEnum getDoubleHeight()

- public void setDoubleHeight(SettingEnum doubleHeight)
  For ESC, Pin
  double height setting
  Parameters:
  doubleHeight - Enable, Disable

- public int getFontStyle()

- public void setFontStyle(int fontStyle)
  For Pin
  Font Style setting
  Parameters:
  fontStyle - FONT_STYLE_NORMAL, FONT_STYLE_HOLLOW, FONT_STYLE_SHADOW, FONT_STYLE_HOLLOW_AND_SHADOW

- public int getPinPrintMode()

- public void setPinPrintMode(int pinPrintMode)
  For Pin
  Print Mode setting
  Parameters:
  pinPrintMode - PIN_PRINT_MODE_Bidirectional, PIN_PRINT_MODE_ONEWAY, PIN_PRINT_MODE_Full_Bidirectional

- public SettingEnum getDoubleSizeChineseCharctor()

- public void setDoubleSizeChineseCharctor(SettingEnum doubleSizeChineseCharctor)
  Deprecated.
  Deprecated, No Working
  For Pin
  Chinese charactor setting
  Parameters:
  doubleSizeChineseCharctor - accepted value: CommonEnum.Setting.Enable/CommonEnum.Setting.Disable

- public SettingEnum getIsAntiWhite()

- public void setIsAntiWhite(SettingEnum isAntiWhite)
  For ESC,XSim
  anti white setting
  Parameters:
  isAntiWhite - SettingEnum.Enable = AntiWhite, SettingEnum.Disable = Cancel

- public SettingEnum getIsEscSmallCharactor()

- public void setIsEscSmallCharactor(SettingEnum isEscSmallCharactor)
  For ESC
  small charactor setting
  Parameters:
  isEscSmallCharactor - SettingEnum.Enable = small font,Font-B), SettingEnum.Disable =Normalfont,Font-A

- public Position getTxtPrintPosition()
  For Tsc, Cpcl,Zpl
  Get text print position object
  Returns:
  Position Object

- public void setTxtPrintPosition(Position txtPrintPosition)
  For Tsc,Cpcl,Zpl
  set text print position
  Parameters:
  txtPrintPosition - position, Default: x=0,y=0

- public TscFontTypeEnum getTscFontTypeEnum()

- public void setTscFontTypeEnum(TscFontTypeEnum tscFontTypeEnum)
  For Tsc
  Tsc Font Type Style
  Parameters:
  tscFontTypeEnum -

- public int getxMultiplication()

- public void setxMultiplication(int xMultiplication)
  For Tsc, Cpcl, Zpl
  set Font X-Multiplication
  Parameters:
  xMultiplication -

- public int getyMultiplication()

- public void setyMultiplication(int yMultiplication)
  For Tsc, Cpcl, Zpl
  set Font Y-Multiplication
  Parameters:
  yMultiplication -

- public PrintRotation getPrintRotation()

- public void setPrintRotation(PrintRotation printRotation)
  For Tsc
  Print rotation setting
  For tsc text, barcode print
  Parameters:
  printRotation - Rotate0, Rotate90, Rotate180, Rotate270

- public CpclFontTypeEnum getCpclFontTypeEnum()

- public void setCpclFontTypeEnum(CpclFontTypeEnum cpclFontTypeEnum)
  Cpcl Font Type Style
  Parameters:
  cpclFontTypeEnum -

- public int getCpclFontSize()

- public void setCpclFontSize(int cpclFontSize)
  For Cpcl
  cpcl text size setting, only support english and number
  Parameters:
  cpclFontSize -

- public ZplFontTypeEnum getZplFontTypeEnum()

- public void setZplFontTypeEnum(ZplFontTypeEnum zplFontTypeEnum)
  For Zpl, Font type Setting
  Parameters:
  zplFontTypeEnum - 0=Vector font (downloadable font),Other resident fonts:A~G,1~9

- public int getZplWidthFactor()

- public void setZplWidthFactor(int zplWidthFactor)
  For Zpl
  width factor setting
  Parameters:
  zplWidthFactor -
  accept value:
  Vector font(FONT_DOWNLOAD_FONT): 10~32000,
  Dot matrix font(Font A~G, 1~9): 1~10

- public int getZplHeightFactor()

- public void setZplHeightFactor(int zplHeightFactor)
  For Zpl
  height factor setting
  Parameters:
  zplHeightFactor -
  accept value:
  Vector font(FONT_DOWNLOAD_FONT): 10~32000,
  Dot matrix font(Font A~G, 1~9): 1~10

- public ESCFontTypeEnum getEscFontType()

- public ESCPFontTypeEnum getEscpFontType()

- public void setEscFontType(ESCFontTypeEnum escFontType)
  For ESC Portable Printer,
  Font Type Setting, Only support English-Number charactor
  Parameters:
  escFontType - Accept Value[ESCFontTypeEnum]:
  FONT_A_12x24
  FONT_B_9x24
  FONT_C_9x17
  FONT_D_8x16

- public void setEscpFontType(ESCPFontTypeEnum e)

- public void setPrintContent(java.lang.String e)

- Public String getPrintContent()

- public int getCpclTextSpacing()

- public void setCpclTextSpacing(int cpclTextSpacing)
  For CPCL
  text spacing setting.
  Parameters:
  cpclTextSpacing - n >= 0

- public XSimFontTypeEnum getxSimFontTypeEnum()

- public void setxSimFontTypeEnum(XSimFontTypeEnum xSimFontTypeEnum)
  zac add for XSim font
  Parameters:
  xSimFontTypeEnum - FONT_0~FONT_15

BarcodeSetting

- public BarcodeStringPosition getBarcodeStringPosition()
- public BarcodeType getBarcodeType()
- public void setBarcodeType(BarcodeType barcodeType)
- public void setBarcodeStringPosition(BarcodeStringPosition barcodeStringPosition)
  For ESC,Tsc
  Set the bar code content location
  Parameters:
  barcodeStringPosition - NONE = [Hidden], ABOVE_BARCODE , BELOW_BARCODE, ABOVE_BELOW_BARCODE
- public EscBarcodePrintOritention getEscBarcodePrintOritention()
- public String getBarcodeContent()
- public void setBarcodeContent(java.lang.String b)
  @Deprecated
- public void setEscBarcodePrintOritention(EscBarcodePrintOritention escBarcodePrintOritention)
  Deprecated.

  Deprecated
  Barcode Oritention Setting
  Parameters:
  escBarcodePrintOritention - Clockwise, Rotate0, Rotate90, Rotate270
- public int getQrcodeDotSize()
- public void setQrcodeDotSize(int qrcodeDotSize)
  For Esc, Tsc
  QRCode dot size settings
  Parameters:
  qrcodeDotSize - Esc(1~15), Tsc(1~10)
- public PrintRotation getPrintRotation()
- public void setPrintRotation(PrintRotation printRotation)
  For Tsc
  barcode rotation settings
  Parameters:
  printRotation -
- public int getHeightInDot()
- public void setHeightInDot(int heightInDot)
  For Tsc,Cpcl
  barcode height settings
  Parameters:
  heightInDot - In Dot
- public int getNarrowInDot()

- public void setNarrowInDot(int narrowInDot)
  For Tsc
  narrow bar settings
  Parameters:
  narrowInDot - In Dot

- public int getWideInDot()

- public void setWideInDot(int wideInDot)
  For Tsc
  wide bar settings
  Parameters:
  wideInDot - In Dot

- public Position getPosition()

- public void setPosition(Position position
  For Tsc, Cpcl, Zpl
  Set the bar code printing position
  Parameters:
  position - position.x , position.y

- public QrcodeEccLevel getQrcodeEccLevel()

- public void setQrcodeEccLevel(QrcodeEccLevel qrcodeEccLevel)
  Error correction capability level
  ():L=7%,M=15%,Q=25%,H=30%
  Parameters:
  qrcodeEccLevel - L=7%,M=15%,Q=25%,H=30%

- public int getBarcodeWidth()

- public void setBarcodeWidth(int barcodeWidth)
  For Esc
  Bar code width, range: 2~6
  barcode width, range: 2~6, default:3
  Parameters:
  barcodeWidth - range: 2~6, default:3
- public void setEscBarcodFont(ESCBarcodeFontTypeEnum font)

- public ESCBarcodeFontTypeEnum getEscBarcodFont()

- public int getAZTEC_xd()

- public void setAZTEC_xd(int AZTEC_xd)

- public int getAZTEC_ec()

- public void setAZTEC_ec(int AZTEC_ec)

- public int getTsc_alignment()

- public void setTsc_alignment(int tsc_alignment)
  20220630 TSPL2 alignment Specify the alignment of barcode 0 : default (Left) 1 : Left 2 : Center 3 : Right

- public int getHuman_readable()

- public void setHuman_readable(int human_readable)
  20220630 TSPL2 human readable 0: not readable 1: human readable aligns to left 2: human readablealigns to center 3: human readable aligns to right

- public int getxSimPdf417EH()

- public void setxSimPdf417EH(int xSimPdf417EH)
  Element Height byte. This is the height length of a single element in the pdf417 symbol, that is, bar/space height. A value between 0x03 to 0x0A can be used. Recommended Value: 0x06
  Parameters:
  xSimPdf417EH -

BitmapSetting

- public void setPrintPostion(@NonNull
  Position printPostion)
  For Tsc, Cpcl

- Bitmap position setting[Position Object], Default: x=0, y=0
  Parameters:
  printPostion -

- public BitmapLimitSizeBean getBitmapLimitSizeBean()

- public void setBitmapLimitSizeBean(BitmapLimitSizeBean bitmapLimitSizeBean)
  Deprecated, No work
  Set the size of the picture to be printed (width, height)
  Parameters:
  bitmapLimitSizeBean - bitmapLimitSizeBean.limitWidth, bitmapLimitSizeBean.limitHeight

- public int getBimtapLimitWidth()

- public void setBimtapLimitWidth(int bimtapLimitWidth)
  Bitmap Limit Width setting(N mm * 8)
  Parameters:
  bimtapLimitWidth -

- public void setBmpPrintMode(BmpPrintMode bmpPrintMode)
  For ESC,TSC
  Bitmap print mode settings
  Parameters:
  bmpPrintMode - BmpPrintMode.MODE_SINGLE_COLOR, BmpPrintMode.MODE_MULTI_COLOR[Default]

EscCmd

- public byte[] getHeaderCmd()
  Description copied from class: Cmd
  Get header commands
  Tsc: CLS command , clear cache
  Esc/Pin: printer initialization

  Specified by:
  getHeaderCmd in class Cmd
  Returns:

- public byte[] getLFCmd()
  Description copied from class: Cmd
  \n
  Wrap

  Specified by:
  getLFCmd in class Cmd
  Returns:

- public byte[] getCRCmd()
  Description copied from class: Cmd
  \r
  Enter

  Specified by:
  getCRCmd in class Cmd
  Returns:

- public byte[] getLFCRCmd()
  Description copied from class: Cmd
  Wrap + Enter

  Specified by:
  getLFCRCmd in class Cmd
  Returns:

- public byte[] getCommonSettingCmd(CommonSetting commonSetting)
  Description copied from class: Cmd
  Common Setting command

  Specified by:
  getCommonSettingCmd in class Cmd
  Returns:

- public byte[] getTextCmd(TextSetting textSetting, java.lang.String text, java.lang.String chartSetName)
  throws java.io.UnsupportedEncodingException
  Description copied from class: Cmd
  Print text command

  Specified by:
  getTextCmd in class Cmd
  Parameters:
  textSetting - Text Setting
  text - Content
  chartSetName - GBK[Default],UTF-8
  Returns:
  Throws:

java.io.UnsupportedEncodingException

- public byte[] getTextCmd(TextSetting textSetting, java.lang.String text)
  throws java.io.UnsupportedEncodingException
  Description copied from class: Cmd
  Print text command

  Specified by:
  getTextCmd in class Cmd
  Parameters:
  textSetting - text set
t ext - chartsetname setting, default:GBK
  Returns:
  Throws:
  java.io.UnsupportedEncodingException

- public byte[] getBitmapCmd(BitmapSetting bitmapSetting, Bitmap bitmap)
  Description copied from class: Cmd
  Get Bitmap printing command

  Specified by:
  getBitmapCmd in class Cmd
  Parameters:
  bitmapSetting - Bitmap setting (width limit)
  bitmap - the bitmap which to print
  Returns:
  byte[] cmds

- public byte[] getBarcodeCmd(BarcodeType barcodeType, BarcodeSetting barcodeSetting, java.lang.String barcode-Content)
  throws SdkException
  Description copied from class: Cmd
  Get Barcode Printing command

  Specified by:
  getBarcodeCmd in class Cmd
  Parameters:
  barcodeType - Bar code type
  barcodeSetting - Bar code setup
  barcodeContent - Barcode content
  Returns:
  Throws:
  SdkException

- public byte[] getSelfTestCmd()
  Description copied from class: Cmd
  Test page print

  Specified by:
  getSelfTestCmd in class Cmd
  Returns:

- public byte[] getBeepCmd()
  Description copied from class: Cmd
  For Esc
  Beep

  Specified by:
  getBeepCmd in class Cmd
  Returns:

- public byte[] getAllCutCmd()
  Description copied from class: Cmd
  For Esc
  All cut

  Specified by:
  getAllCutCmd in class Cmd
  Returns:

- public byte[] getDrawLine(int x0, int y0, int x1, int y1, int lineWidth)
  Description copied from class: Cmd
  For TSC, CPCL, ZPL
  Draw a line.[only support Horizontal lines and straight lines].

  Specified by:
  getDrawLine in class Cmd
  Parameters:
  x0 - start coordinate-x
  y0 - start coordinate-y
  x1 - end coordinate-x
  y1 - end coordinate-y
  lineWidth - line width in Dot.
  Returns:

- public byte[] getDrawBox(int x0, int y0, int x1, int y1, int lineWidth)
  Description copied from class: Cmd
  Draw Box.[For TSC, CPCL, ZPL]

  Specified by:
  getDrawBox in class Cmd
  Parameters:
  x0 - left-top of box coordinate-x
  y0 - left-top of box coordinate-y
  x1 - right-bottom of box coordinate-x
  y1 - right-bottom of box coordinate-y
  lineWidth - the line width of box, in dot.
  Returns:

- public byte[] getPrinterStatus(PrinterAskStatusEnum printerAskStatusEnum)
  Get printer status
  Parameters:
  printerAskStatusEnum - 1: n = 1: transfer printer status,
  n = 2: transfer offline status,
  n = 3: transfer error status,
  n = 4: transfer roll sensor status
  Returns:

- public byte[] getSetLeftStartSpacing(java.lang.Integer n)
  Set the amount of left margin
  Parameters:
  n - 1mm = n*8
  Returns:

- public byte[] getSetAreaWidth(java.lang.Integer n)
  Set the print area width
  Parameters:
  n - 1mm = n*8
  Returns:

- public byte[] getReverse(int x0, int y0, int x1, int y1)
  Description copied from class: Cmd
  For TSC
  Function: Invert the specified area in the image buffer

  Specified by:
  getReverse in class Cmd
  Parameters:
  x0 - start coordinate-x
  y0 - start coordinate-y
  x1 - end coordinate-x
  y1 - end coordinate-y
  Returns:

- public byte[] getSetXPosition(java.lang.Integer n)
  Set absolute print position (x-axis position)
  Parameters:
  n - Points 1mm = n*8
  Returns:

- public byte[] getPageMode(boolean isPageMode)
  Set page mode or standard mode. true: page mode, false: standard mode
  Parameters:
  isPageMode -
  Returns:

- public byte[] getPageEnd(boolean isRtnStandard)
  Print page schema data
  Parameters:
  isRtnStandard - true:Back to standard mode, false: Not going back to standard mode
  Returns:

- public byte[] getPageArea(int x0, int y0, int x1, int y1)
  In page mode, set the print area
  Parameters:
  x0 - Starting position The position of the x axis
  y0 - Starting position Position of the y axis
  x1 - End position Position of the X-axis
  y1 - End position Position of the Y-axis
  Returns:

Code : 0577200M000087 Rev. 1.00

CUSTOM S.p.A.
World Headquarters
Via Berettine, 2/B - 43010 Fontevivo, Parma ITALY
Tel. +39 0521 680111 - Fax +39 0521 610701
info@custom.biz - www.custom.biz

All rights reserved

www.custom.biz