Histone Service Programming Interface

2019.09

1、 Purpose 6

2、 Terminology....6
2.1 Service....6
2.2 Android Interface definition language....6
2.3 Service Application....6
3、 constants and definitions....6
4、 service program overview....6
5、 device service engine module statement....7
5.1 Module Description....7
5.2 package path....7
5.3 File name....7
5.4 AIDL Interface list....7
5.4.1 Get the Service Version Number (getSDKVersion)....8
5.4.2 Configure service (setServiceConifg)....8
5.4.3 Get a Service Object (getService)....8
6、 printer module statement....9
6.1 Module Description....9
6.2 package path....9
6.3 AIDL file name....9
6.4 Interface List....9
6.4.1 Printer Configuration (setConfig)....9
6.4.2 Get Status (checkStatus)....10
6.4.3 Open Cash drawer(openCashBox)....10
6.4.4 Get Printer Description (getDescriptions)....10
6.4.5 Extended command Set (addCommand)....11
6.4.6 Paper cutting (addCut)....11
6.4.7 Feed paper by line (addFeedLine)....11
6.4.8 Feed Paper by pixel (addFeedDots)....12
6.4.9 Double font size (addTextSizeDouble)....12
6.4.10 Normal font size (addTextSizeNormal)....12
6.4.11 Print text(addText)....13
6.4.12 Print texts in table form(addTextsAutoWrap)....13
6.4.13 Set Alignment (setAlign)....13
6.4.14 Print image(addImage)....14
6.4.15 Print QR code (addQRCode)....14
6.4.16 Print barcode (addBarcode)....14
6.4.17 Start building a receipt layout (beginTranscation)....15
6.4.18 End building a receipt layout (endTranscation)....15
6.4.19 Start printing (commit)....15
6.4.20 Register a listener for printer status changes(registerStatusListener)....16
6.4.21 Unregister the listener for printer status changes(unRegisterStatusListener)....16
7、 Mainboard cash drawer module statement....17
7.1 module statement....17
7.2 Package path....17
7.3 AIDL file name....17

7.4 Interface list 17
7.4.1 Open cash drawer (openCashBox) 17
7.4.2 Get cash drawer status (getCashBoxStatus) 17
8、 MSR card reader module statement 18
8.1 module statement 18
8.2 package path 18
8.3 AIDL file name 18
8.4 Interface list 18
8.3.1 MSR Initialization (openMsr) 18
8.3.2 Set timeout (setTimeout) 18
8.3.3 Register a MSR Listener (setMsrListener) 19
8.3.4 stop card detection (stopCheck) 19
8.3.5 Release MSR resources (closeMsr) 19
8.5 MSR event listener 20
9、 Scanner module statement 20
9.1 module statement 20
9.2 package path 20
9.3 AIDL file name 20
9.4 Interface list 20
9.4.1 Initialize scanner (initScanner) 21
9.4.2 Destroy scanner resources (destoryScanner) 21
9.4.3 Set scanner data listener (setScanListener) 21
9.4.4 Cancel scanner data listener (removeScanListener) 22
10、 RFID module statement 22
10.1 module statement 22
10.2 package path 22
10.3 AIDL file name 22
10.4 Interface list 22
10.4.1 RFID initialization (piccOpen) 23
10.4.2 RFID resource release (piccClose) 23
10.4.3 Set timeout(setTimeout) 23
10.4.4 Register a RFID event listener (setPiccListener) 24
10.4.5 Stop card detection(stopCheck) 24
10.4.6 MIFARE CARD Authentication (piccMfAuthBlock) 24
10.4.7 Mifare card Reading data (piccMfReadBlock) 25
10.4.8 Mifare card writing data(piccMfWriteBlock) 25
10.4.9 Mifare card Initial value (piccMfInitValue) 26
10.4.10 Mifare card Added value (piccMfAddValue) 26
10.4.11 Mifare card Impairment value (piccMfReduceValue) 26
10.5 Events listener 27
11、 Serial port module statement 27
11.1 module statement 27
11.2 package path 27
11.3 AIDL file name 27

11.4 Interface list 27
11.4.1 Open serial port device (openUart) 27
11.4.2 Close serial port devices (closeUart) 28
11.4.3 Read serial port data (readUart) 28
11.4.4 Write serial port data (writeUart) 28
11.4.5 Clear serial port data (flushUart) 29
11.4.6 Set serial port baud rate (setBaudateRate) 29
12、 Bagging machine module statement 29
12.1 module statement 29
12.2 package path 29
12.3 AIDL file name 30
12.4 Interface list 30
12.4.1 Auto bag output (autoOutBag) 30
12.4.2 Reset the number of historical bags output(clearBaggerHistoryTotal) 30
12.4.3 Get the historical output number (getBaggerHistoryTotal) 31
12.4.4 Get the status of the bag output channel (getBaggerStatus) 31
12.4.5 Set the bag output Channel Listener (setBaggerListener) 31
13、 Indicator light module statement 33
15.1 module statement 33
15.2 package path 33
15.3 AIDL file name 33
15.4 Interface list 33
13.4.1 Lighting indicator (setLight) 33
13.4.2 Get the indicator light color (getLightStatus) 34
13.4.3 Control buzzer (buzzerCtrl) 34
14、 physical button module statement 34
14.1 module statement 35
14.2 package path 35
14.3 AIDL file name 35
14.4 Interface list 35
15.1.1 Get button light status (getKeyLightStatus) 35
15.2.1 Set button light status (setKeyLight) 35
15.3.1 Set button event listener (setKeyListener) 36
15、 Scale module statement 36
15.1 Scale Module statement 37
15.2 Package path 37
15.3 AIDL file name 37
15.4 Interface list 37
15.4.1 Open scale(open) 37
15.4.2 Close scale(close) 37
15.4.3 Get weight(getWeight) 37
15.4.4 Set zero weight(setZeroWeight) 38
15.4.5 Tare weight(setTareWeight) 38
15.4.1 Config scale(config) 39

15.5 Event Listener.....39

Version control information

version	Date	stage	Participant	Notes
		Draft		

Document review information

version	Reviewer	Role	Review date	Confirmation（name）	Notes

Document approval information

version	Approver	Role	Approval date	Confirmation（name）	Notes

1、 Purpose

In order to integrate enterprise resources and improve the efficiency of POS terminal software development, this specification has been specially formulated.

2、 Terminology

2.1 Service

One of the four major components in the Android system

This component is almost identical to the level of the Activity, but can only run in the background and can interact with other components.

The following ways can start the service (Service)：

（Context）ctx.bindService()

2.2 Android Interface definition language

Since the processes in the Android system cannot share memory, there is a need to provide some mechanism for data communication between different processes.

In order to enable other applications can also access services provided by this application, Android system uses a Remote Procedure Call (Remote Procedure Call, RPC) to achieve this purpose.

2.3 Service Application

A service application that follows the service layer interface of this specification, which is developed by Hisense. It can drive Hisense hardware devices (magnetic strip or non-card reader, buzzer, etc.), which is integrated into the POS product by default for customer development calls.

3、 constants and definitions

The serial number of the external peripheral and the serial port name of the printer are defined. If the following interface does not meet the requirements, it can be packaged according to the development documentation.

The device nodes of the serial port are: /dev/ttySAC0, /dev/ttySAC3

4、 service program overview

service name:

Com.histonepos.psdk

Instructions:

By binding com.histonepos.psdk service, obtain an instance returned after a successful connection, you can use all the interface services programs.

Examples are as follows, for reference only：

```
SmartService.getInstance().register(this, new IServiceConnector() {
    @Override
    public void onServiceDisconnected() {
        // TODO Auto-generated method stub
        GlobalDefines.cashbox = null;
    }

    @Override
    public void onServiceConnected(IServiceManager serviceManager) {
        // TODO Auto-generated method stub
        Log.i("PosPad_Activity", "onServiceConnected");
        GlobalDefines.sm    = serviceManager;
    }
});
```

Release service resources when the app exits:

```
SmartService.getInstance().unregister();
```

5、 device service engine module statement

5.1 Module Description

Used to manage all built-in device interfaces, such as MSR card reader, printers, RFID card reader, scanners, cash drawers, etc.

5.2 package path

com.histonepos.npsdk.api

5.3 File name

IServiceManager.aidl

5.4 AIDL Interface list

Interface prototype	Function Description	Description
getSdkVersion	Get the service version	
setServiceConfig	Configure service	Reserved

getService	Get the service object	

5.4.1 Get the Service Version Number (getSDKVersion)

Interface prototype	String getSDKVersion()	
Interface function	Get the version number	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	success version number eg.V1.00.J509 failed Return NULL	
Notes		

5.4.2 Configure service (setServiceConifg)

Interface prototype	Void setServiceConfig(in android.os.Bundle extra)	
Interface function	Configure service	
Parameter Description	Input Parameter	Data
	Output Parameter	Null
return value	Null	
Notes	Interface reserved	

5.4.3 Get a Service Object (getService)

Interface prototype	IBinder getService(int sType);	
Interface function	Return the object of each module service	
Parameter Description	Input Parameter	sType：Refer to Const constant definition
	Output Parameter	Null
return value	success Returns the service module object corresponding to sType failed Return NULL	
Notes		

6、 printer module statement

6.1 Module Description

Used to operate printer devices.

6.2 package path

com.histonepos.npsdk.printer

6.3 AIDL file name

IPrinterService.aidl

6.4 Interface List

Interface prototype	Function Description
Void setConfig(Bundle data)	Printer configuration
Int checkStatus()	Get status
Void openCashBox()	Open the cash drawer
Bundle getDescriptions();	Get device description
Void addCommand(byte[] data)	Extended command set
Void addCut()	Cut paper
void addFeedLine(int line);	Line paper feed
void addFeedDots(int dots);	Pixels paper feed
void addTextSizeDouble();	Double font
void addTextSizeNormal();	Normal font
void addText(String data, int pattern);	Print text
void addImage(in Bitmap data);	Print picture
void addQRCode(String qrcode,int mode);	Print QR
void addBarcode(String barcode);	Print barcode
void beginTranscation();	Start building a receipt layout
int endTranscation();	End building a receipt layout
int commit(int transId);	Submit print
void registerStatusListener(IPrinterStatusListener statusChange);	Register a listener for printer status changes
void unRegisterStatusListener();	Unregister the listener for printer status changes

6.4.1 Printer Configuration (setConfig)

Note：the process of building a receipt must be put in a new thread.

Interface prototype	Void setConfig(Bundle data)

Interface function	Printer configuration	
Parameter Description	Input Parameter	Data：Set the printer width, etc.
	Output Parameter	Null
return value	Null	
Notes		

6.4.2 Get Status (checkStatus)

Interface prototype	Int checkStatus()	
Interface function	Get printer status	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	Normal positive value (>=0) Abnormal negative value (<0), refer to the constant in PrinterConsts	
Notes		

6.4.3 Open Cash drawer(openCashBox)

Interface prototype	Void openCashBox()	
Interface function	Open Cash drawer	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	Null	
Notes		

6.4.4 Get Printer Description (getDescriptions)

Interface prototype	Bundle getDescriptions();
Interface	Get Printer Description

function		
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	Bundle	
Notes	reserved	

6.4.5 Extended command Set (addCommand)

Interface prototype	Void addCommand(byte[] data)	
Interface function	Extended command Set	
Parameter Description	Input Parameter	Data:command
	Output Parameter	Null
return value	Null	
Notes		

6.4.6 Paper cutting (addCut)

Interface prototype	Void addCut()	
Interface function	Paper cutting	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	Null	
Notes		

6.4.7 Feed paper by line (addFeedLine)

Interface prototype	void addFeedLine(int line);
Interface function	Feed paper by line

Parameter Description	Input Parameter	Line：Number of rows
	Output Parameter	Null
return value	Null	
Notes		

6.4.8 Feed Paper by pixel (addFeedDots)

Interface prototype	void addFeedDots(int dots);	
Interface function	Feed paper by dots	
Parameter Description	Input Parameter	Dots:paper feeding dots number
	Output Parameter	Null
return value	Null	
Notes		

6.4.9 Double font size (addTextSizeDouble)

Interface prototype	void addTextSizeDouble();	
Interface function	Double font size	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	Null	
Notes		

6.4.10 Normal font size (addTextSizeNormal)

Interface prototype	void addTextSizeNormal();	
Interface function	normal font size	
Parameter	Input	Null

Description	Parameter	
	Output Parameter	Null
return value	Null	
Notes		

6.4.11 Print text(addText)

Interface prototype	void addText(String data, int pattern);	
Interface function	Print text	
Parameter Description	Input Parameter	Data: character pattern: alignment mode 0 left alignment 1 center 2 right alignment
	Output Parameter	Null
return value	Null	
Notes		

6.4.12 Print texts in table form(addTextsAutoWrap)

Interface prototype	void addTextsAutoWrap(in int[] weights, in int[] aligns, in String[] datas);	
Interface function	Print texts in table form	
Parameter Description	Input Parameter	Weights：weights for each column aligns：alignments for each column Datas：contents for each column
	Output Parameter	Null
return value	Null	
Notes	weights must be the simplest, the sum of the elements of weights must be 2 or 4 or 8,otherwise IllegalArgumentException will be thrown.	

6.4.13 Set Alignment (setAlign)

Interface prototype	void setAlign(in int align);
Interface	Set alignment for bitmap,barcode,qr

function		
Parameter Description	Input Parameter	Align 0:left 1：centre 2：right please refer to demo
	Output Parameter	Null
return value	Null	
Notes		

6.4.14 Print image(addImage)

Interface prototype	void addImage(in Bitmap data);	
Interface function	Print image	
Parameter Description	Input Parameter	Data:bitmap
	Output Parameter	Null
return value	Null	
Notes	If not set alignment by calling setAlign(),the default alignment is to centre	

6.4.15 Print QR code (addQRCode)

Interface prototype	void addQRCode(String qrcode,int mode);	
Interface function	Print QR code	
Parameter Description	Input Parameter	qrcode: content mode:zoom mode
	Output Parameter	Null
return value	Null	
Notes	If not set alignment by calling setAlign(),the default alignment is to centre	

6.4.16 Print barcode (addBarcode)

Interface prototype	void addBarcode(String barcode);
Interface function	Print barcode

Parameter Description	Input Parameter	barcode: barcode content
	Output Parameter	Null
return value	Null	
Notes	If not set alignment by calling setAlign(),the default alignment is to centre	

6.4.17 Start building a receipt layout (beginTranscation)

Interface prototype	void beginTranscation();	
Interface function	Start building a receipt layout	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	Null	
Notes	The receipt generation process must follow the specified calling sequence. To ensure code execution adheres to this sequence, multithreading is not supported in receipt generation. 1. bengingTranscation() 2. addXXX() 3. endTranscation() 4.commit()	

6.4.18 End building a receipt layout (endTranscation)

Interface prototype	int endTranscation();	
Interface function	End building a receipt layout	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	Return transId, Mark document ID	
Notes		

6.4.19 Start printing (commit)

Interface prototype	int commit(int transId);	
Interface function	Start printing	
Parameter Description	Input Parameter	transId - the printed content document is marked, returned by endTranscation()
	Output Parameter	Zero for sending success or negative value for failure
return value	Null	
Notes		

6.4.20 Register a listener for printer status changes(registerStatusListener)

Interface prototype	void registerStatusListener(IPrinterStatusListener statusChange);	
Interface function	Register a listener for printer status changes	
Parameter Description	Input Parameter	statusChange: Callback the function
	Output Parameter	Null
return value	Null	
Notes		

6.4.21 Unregister the listener for printer status changes(unRegisterStatusListener)

Interface prototype	void unRegisterStatusListener();	
Interface function	Unregister the listener for printer status changes	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	Null	
Notes		

7、 Mainboard cash drawer module statement

7.1 module statement

Used for operational cash box equipment.

7.2 Package path

com.histonepos.npsdk.cashbox

7.3 AIDL file name

ICashBoxService.aidl

7.4 Interface list

Interface prototype	Function Description
int openCashBox(int id);	Open cash drawer
int getCashBoxStatus();	Get cash drawer status

7.4.1 Open cash drawer (openCashBox)

Interface prototype	int openCashBox(int id);	
Interface function	Open cash drawer	
Parameter Description	Input Parameter	Id:1:：cash drawer 1　　2：cash drawer 2
	Output Parameter	Null
return value	success　　0 failed　　Negative value	
Notes		

7.4.2 Get cash drawer status (getCashBoxStatus)

Interface prototype	int getCashBoxStatus();	
Interface function	Get cash drawer status	
Parameter Description	Input Parameter	Null
	Output Parameter	Null

return value	success 0 cash drawer open 1：cash drawer close failed Negative value
Notes	Non-printer drive cash drawer interface

8、 MSR card reader module statement

8.1 module statement

used for managing MSR card reader

8.2 package path

com.histonepos.npsdk.msr

8.3 AIDL file name

IMsrService.aidl

8.4 Interface list

Interface prototype	Function Description
int openMsr()	MSR initialization
void setTimeout(int t);	Checking card timeout ,time in seconds
void setMsrListener(IMsrEventListener listener);	Register a MSR listener and start checking cards
void stopCheck();	Not in the timeout state, terminate detection
int closeMsr();	Release MSR resources

8.3.1 MSR Initialization (openMsr)

Interface prototype	int openMsr()	
Interface function	MSR Initialization	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	success 0 failed Negative value	
Notes		

8.3.2 Set timeout (setTimeout)

Interface	void setTimeout(int t);

prototype		
Interface function	Set timeout	
Parameter Description	Input Parameter	t Unit seconds
	Output Parameter	Null
return value	Null	
Notes		

8.3.3 Register a MSR Listener (setMsrListener)

Interface prototype	void setMsrListener(IMsrEventListener listener);	
Interface function	Register a MSR listener and start checking cards	
Parameter Description	Input Parameter	listener： MSR callback
	Output Parameter	Null
return value	Null	
Notes		

8.3.4 stop card detection (stopCheck)

Interface prototype	void stopCheck();	
Interface function	stop card detection	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	Null	
Notes	Not in the timeout state, stop card detection	

8.3.5 Release MSR resources (closeMsr)

Interface prototype	int closeMsr();	
Interface function	Release MSR resources	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	success 0 failed Negative value	
Notes		

8.5 MSR event listener

```
IMsrEventListener {

/*respCode success =    0, Other values failed

  Bundle save when success
KEY (String): "1" or "2" or "3"
For example, the first track trackResult .getByteArray("1")
              */
void onTrack(int respCode,in android.os.Bundle trackResult);

   /**
     * detect card timeout
    */
    void onTimeout();
}
```

9、 Scanner module statement

9.1 module statement

used for managing scanner

9.2 package path

com.histonepos.npsdk.scanner

9.3 AIDL file name

IScannerService.aidl

9.4 Interface list

Interface prototype	Function Description

int initScanner();	Initialize scanner
int destoryScanner();	Destroy scanner resources
void setScanListener(IScanEventListener listener);	Set scanner data listener
void removeScanListener();	Cancel scanner data listener

9.4.1 Initialize scanner (initScanner)

Interface prototype	int initScanner();	
Interface function	Initialize scanner	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	success >= 0 failed < 0	
Notes		

9.4.2 Destroy scanner resources (destoryScanner)

Interface prototype	int destoryScanner();	
Interface function	Destroy scanner resources	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	success >= 0 failed < 0	
Notes		

9.4.3 Set scanner data listener (setScanListener)

Interface prototype	void setScanListener(IScanEventListener listener);	
Interface function	Set scanner data listener	
Parameter Description	Input Parameter	listener - Button event callback
	Output Parameter	Null

return value	Null
Notes	

9.4.4 Cancel scanner data listener (removeScanListener)

Interface prototype	void removeScanListener();	
Interface function	Cancel scanner data listener， No longer accepting scanned data	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	Null	
Notes		

10、 RFID module statement

10.1 module statement

used for managing device RFID module

10.2 package path

com.histonepos.npsdk.picccard

10.3 AIDL file name

IPiccCardService.aidl

10.4 Interface list

Interface prototype	Function Description
int piccOpen();	RFID initialization
void piccClose();	RFID resource release
void setTimeout(int t);	Set timeout
void stopCheck();	Not in the timeout state, stop card detection
Void setPiccListener(IPiccEventListener listener);	Register a RFID listener and start checking cards
int piccMfAuthBlock(byte type, byte blockaddr,in byte[] keydata, in byte[]	Certification

serialInfo);	
int piccMfReadBlock(byte blockaddr, inout BytesArray blockdata);	Reading data
int piccMfWriteBlock(byte blockaddr, in byte[] data);	Writing data
int piccMfInitValue(byte blockaddr, int value);	Initial value
int piccMfAddValue(byte blockaddr, int value);	Added value
int piccMfReduceValue(byte blockaddr, int value);	Impairment value

10.4.1 RFID initialization (piccOpen)

Interface prototype	int piccOpen();	
Interface function	RFID initialization	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	success 0 failed Negative value	
Notes	Note! Initialization corresponds to piccClose, and piccClose is required after finished.	

10.4.2 RFID resource release (piccClose)

Interface prototype	void piccClose();	
Interface function	RFID resource release	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	Null	
Notes	Note! Initialization corresponds to piccClose, and piccClose is required after finished.	

10.4.3 Set timeout(setTimeout)

Interface prototype	void setTimeout (int t);	
Interface function	Checking card timeout ,time in seconds	
Parameter Description	Input Parameter	t seconds
	Output Parameter	Null
return value	Null	
Notes		

10.4.4 Register a RFID event listener (setPiccListener)

Interface prototype	void setPiccListener (IPiccEventListener listener);	
Interface function	Register a RFID event listener and start checking cards	
Parameter Description	Input Parameter	listener
	Output Parameter	Null
return value	Null	
Notes		

10.4.5 Stop card detection(stopCheck)

Interface prototype	void stopCheck ();	
Interface function	Stop card detection	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	Null	
Notes		

10.4.6 MIFARE CARD Authentication (piccMfAuthBlock)

Interface	int piccMfAuthBlock(byte type, byte blockaddr,in byte[] keydata, in byte[]

prototype	serialInfo1);	
Interface function	Authentication	
Parameter Description	Input Parameter	@param type Authentication type, using KEYA, KEYB Value definition: 0x00: KEYA 0x01: KEYB @param blockaddr Block number to be authenticated @param keydata value (informed in clear text) 6bytes @param serialInfo1 Reset information (device card number, no length information)
	Output Parameter	Null
return value	success 0 failed others	
Notes		

10.4.7 Mifare card Reading data (piccMfReadBlock)

Interface prototype	int piccMfReadBlock(byte blockaddr, inout BytesArray blockdata);	
Interface function	Reading data	
Parameter Description	Input Parameter	@param blockaddr Block number
	Output Parameter	@param blockdata Data returned on success, a total of 16 bytes of data
return value	success 0 failed others	
Notes		

10.4.8 Mifare card writing data(piccMfWriteBlock)

Interface prototype	int piccMfWriteBlock(byte blockaddr, in byte[] data);	
Interface function	Wrinting data	
Parameter Description	Input Parameter	@param blockaddr Block number @param data To be written data, it must be 16 bytes of data. If the number of data bits is insufficient, the internal needs to be complemented by zero 0x00.
	Output Parameter	Null
return value	success 0 failed others	

Notes	

10.4.9 Mifare card Initial value (piccMfInitValue)

Interface prototype	int piccMfInitValue (byte blockaddr, int value);	
Interface function	Initial value	
Parameter Description	Input Parameter	@param blockaddr Block number @param value
	Output Parameter	Null
return value	success 0 failed others	
Notes		

10.4.10 Mifare card Added value (piccMfAddValue)

Interface prototype	int piccMfAddValue(byte blockaddr, int value);	
Interface function	Added value	
Parameter Description	Input Parameter	@param blockaddr Block number @param value
	Output Parameter	Null
return value	success 0 failed others	
Notes		

10.4.11 Mifare card Impairment value (piccMfReduceValue)

Interface prototype	int piccMfReduceValue(byte blockaddr, int value);	
Interface function	Impairment value	
Parameter Description	Input Parameter	@param blockaddr Block number @param value
	Output Parameter	Null
return value	success 0 failed others	
Notes		

10.5 Events listener

```
interface IPiccEventListener {
    /**
     * The RFID card is activated successfully.
     */
    void onCardActivate(byte[] serialInfo);

    /**
      * Check card timeout
     */
     void onTimeout();
}
```

11、 Serial port module statement

11.1 module statement

used for managing serial port peripherals.

11.2 package path

com.histonepos.npsdk.uart

11.3 AIDL file name

IUartDriver.aidl

11.4 Interface list

Interface prototype	Function Description
int openUart (int Slot);	Open serial port device
void closeUart (int Slot);	Close serial port devices
int readUart(int Slot,out byte [] value,int len);	Read serial port data
int writeUart(int Slot,in byte [] value,int len);	Write serial port data
int flushUart (int Slot);	Clear serial port data
int setBaudateRate(int port,int baudrate,boolean ctrlOPen);	Set serial port parameters

11.4.1 Open serial port device (openUart)

Interface prototype	int openUart (int Slot);	
Interface function	Open serial port device	
Parameter Description	Input Parameter	Slot: serial port number

	Output Parameter	Null
return value	success Return file descriptor error Return Negative value	
Notes		

11.4.2 Close serial port devices (closeUart)

Interface prototype	void closeUart (int Slot);	
Interface function	Close serial port devices	
Parameter Description	Input Parameter	Slot: serial port number
	Output Parameter	Null
return value	Null	
Notes		

11.4.3 Read serial port data (readUart)

Interface prototype	int readUart(int Slot,out byte [] value,int len);	
Interface function	Read serial port data	
Parameter Description	Input Parameter	Slot： serial port number Len： Expected data length to read
	Output Parameter	Value： data content is read
return value	success The actual data length is read failed Negative value	
Notes		

11.4.4 Write serial port data (writeUart)

Interface prototype	int writeUart(int Slot,in byte [] value,int len);	
Interface function	Write serial port data	
Parameter Description	Input Parameter	Slot： serial port number Value： Data content Len： The length of data that needs to be written

	Output Parameter	
return value	success Actual written data length failed Negative value	
Notes		

11.4.5 Clear serial port data (flushUart)

Interface prototype	int flushUart (int Slot);	
Interface function	Clear serial port data	
Parameter Description	Input Parameter	Slot：serial port number
	Output Parameter	
return value	success 0 failed Negative value	
Notes		

11.4.6 Set serial port baud rate (setBaudateRate)

Interface prototype	Int setBaudateRate(int port,int baudrate,boolean ctrlOPen);	
Interface function	Set serial port baud rate	
Parameter Description	Input Parameter	Port：serial port number Baudrate：Baud rate ctrlOPen：Flow Control
	Output Parameter	
return value	success 0 failed Negative value	
Notes	Use this after openUart	

12、 Bagging machine module statement

12.1 module statement

used for managing auto bagging machine.

12.2 package path

com.histonepos.npsdk.bagger

12.3 AIDL file name

IBaggerService.aidl

12.4 Interface list

Interface prototype	Function Description
boolean autoOutBag(int position, int num);	Auto bag output
boolean clearBaggerHistoryTotal(int position);	The number of historical bags output of Bagging machine is set to 0
int getBaggerHistoryTotal(int position);	Get the historical output number of bag output channel
int getBaggerStatus(int position);	Get the status of the bag output channel
void setBaggerListener(int position, IBaggerEventListener listener);	Sets the callback events of bag output channel

12.4.1 Auto bag output (autoOutBag)

Interface prototype	boolean autoOutBag(int position, int num);	
Interface function	AAuto bag output, subsequent state callback via IBaggerEventListener	
Parameter Description	Input Parameter	position - The serial number of the plastic bag machine bag output channel, 1, 2, 3... num - The number of bags output, greater than 0
	Output Parameter	Null
return value	success true error false	
Notes		

12.4.2 Reset the number of historical bags output(clearBaggerHistoryTotal)

Interface prototype	boolean clearBaggerHistoryTotal(int position);	
Interface function	The number of historical bags output of Bagging machine is set to 0	
Parameter Description	Input Parameter	position - The serial number of the plastic bag machine bag output channel, 1, 2, 3...
	Output Parameter	Null
return value	success true error false	

Notes	Note: This function of second generation bag machine is invalid! Only the first generation bag spreader works ,and the firmware does not support separate resetting for the number of outgoing bags in a certain channel, No matter the position is 1 or 2, both channel data will be cleared at the same time!

12.4.3 Get the historical output number (getBaggerHistoryTotal)

Interface prototype	int getBaggerHistoryTotal(int position);	
Interface function	Get the historical output number of bag output channel	
Parameter Description	Input Parameter	position - The serial number of the plastic bag machine bag output channel, 1, 2, 3...
	Output Parameter	Null
return value	Get the historical output number of specific bag output channel	
Notes	Note: This function of the second generation bag machine is invalid ! Only the first generation bag machine works!	

12.4.4 Get the status of the bag output channel (getBaggerStatus)

Interface prototype	int getBaggerStatus(int position);	
Interface function	Get the status of the bag output channel	
Parameter Description	Input Parameter	position - The serial number of the plastic bag machine bag output channel, 1, 2, 3...
	Output Parameter	
return value	Specify the status of the channel 1 : Normally usable 0 : No shopping bag -1 : Access door is open -2 : Other --- Cannot work abnormally -3 : no bagging machine	
Notes		

12.4.5 Set the bag output Channel Listener (setBaggerListener)

Interface prototype	void setBaggerListener(int position, IBaggerEventListener listener);
Interface	Sets the callback events of bag output channel

<table>
<tr><td>function</td><td colspan="2"></td></tr>
<tr><td>Parameter Description</td><td>Input Parameter</td><td>position - The serial number of the plastic bag machine bag output channel, 1, 2, 3...
listener – bag output event callback, can be null which indicating clear callback
@see IBaggerEventListener</td></tr>
<tr><td></td><td>Output Parameter</td><td></td></tr>
<tr><td>return value</td><td colspan="2">Null</td></tr>
<tr><td>Notes</td><td colspan="2">IBaggerEventListener callback event
/**
* callback when Channel status change
*
* @param newStatus channel new status
* Constants refer to the return of {@link#getBaggerStatus(int)}
* @param msg status detailed description information or reason
*/
void onBaggerStatusChanged(int newStatus, String msg);

/**
* Start a single bag output event
*/
void onBaggerOutputStart();

/**
* Complete a single bag output event
*/
void onBaggerOutputEnd();

/**
* bag output completion event, this is a total event callback, and does not conflict with {@link #onBaggerOutputEnd()}
*
* <pre>
* For example, if a single bag output command is issued or a bag is manually pulled out, it will receive once onBaggerOutputEnd and once onBaggerCharge.
* After a multiple bags output command is sent, it will receive multiple onBaggerOutputEnd, but will only receive once onBaggerCharge
* </pre>
* @param num number of bag output
*/
void onBaggerCharge(int num);</td></tr>
</table>

13、 Indicator light module statement

15.1 module statement

used for managing device indicator module.

15.2 package path

com.histonepos.npsdk.light

15.3 AIDL file name

ILightService.aidl

15.4 Interface list

Interface prototype	Function Description
boolean setLight(String status);	Lighting light color
String getLightStatus();	Get the light color
boolean buzzerCtrl(String status);	Control buzzer

13.4.1 Lighting indicator (setLight)

Interface prototype	boolean setLight(String status);	
Interface function	Lighting light color	
Parameter Description	Input Parameter	Status - the status code of the customized firmware "HM_LIGHT_OFF" - the light is off, "HM_LIGHT_ON_BLUE" : The blue light is on, "HM_LIGHT_ON_RED" : The red light is on, "HM_LIGHT_BLINK_BLUE" : The blue light flashes. "HM_LIGHT_BLINK_RED" : The red light flashes. Status - the status code of the normal firmware "NORMAL_LIGHT_ON_ORANGE" : The orange light is on, "NORMAL_LIGHT_ON_RED" : The red light is on, "NORMAL_LIGHT_ON_GREEN" : The green light is on, "NORMAL_LIGHT_OFF_ORANGE" : Orange light is off, "NORMAL_LIGHT_OFF_RED" : Red light is off, "NORMAL_LIGHT_OFF_GREEN" : Green light is off, "NORMAL_LIGHT_BLINK_ORANGE" : The orange light flashes,

		"NORMAL_LIGHT_BLINK_RED" : The red light flashes, "NORMAL_LIGHT_BLINK_GREEN" : The green light flashes,
	Output Parameter	Null
return value	success true error false	
Notes	The lights with the customized firmware are as follows: • 1. The flashing interval is 400ms. • 2. The instructions are mutually exclusive. For example, the current blue light is on, when it receives "ON_RED", it switches to red light.	

13.4.2 Get the indicator light color (getLightStatus)

Interface prototype	String getLightStatus();	
Interface function	Get the light color	
Parameter Description	Input Parameter	Null
	Output Parameter	Null
return value	Ceiling light status code, refer to setLight(java.lang.String) parameter	
Notes	No support yet	

13.4.3 Control buzzer (buzzerCtrl)

Interface prototype	boolean buzzerCtrl(String status);	
Interface function	Control buzzer	
Parameter Description	Input Parameter	Status - status code "BUZZER_ON" buzzer sounds "BUZZER_OFF" turns off the buzzer
	Output Parameter	Null
return value	success true error false	
Notes	Buzzer are only available for normal firmware lights. Customized firmware lights do not support this feature.	

14、 physical button module statement

14.1 module statement

used for managing device physical button module.

14.2 package path

com.histonepos.npsdk. physicalkey

14.3 AIDL file name

IPhysicalKeyService.aidl

14.4 Interface list

Interface prototype	Function Description
String getKeyLightStatus(int keyType);	Get button light status
boolean setKeyLight(int keyType, java.lang.String status);	Set button light status
void setKeyListener(int keyType, IEventListener listener);	Set button event listener

15.1.1 Get button light status (getKeyLightStatus)

Interface prototype	String getKeyLightStatus(int keyType);	
Interface function	Get button light status	
Parameter Description	Input Parameter	keyType - the type of button 1 : Help button
	Output Parameter	Null
return value	Light status code, refer to setKeyLight(int, String) status definition	
Notes	No support yet	

15.2.1 Set button light status (setKeyLight)

Interface prototype	boolean setKeyLight(int keyType, java.lang.String status);	
Interface function	Set button light status	
Parameter Description	Input Parameter	keyType - the type of button 1 : Help button Status - button light status "BUTTON_OFF" : The button is off, "BUTTON_ON": The button lights "BUTTON_BLINK": button flashes
	Output Parameter	Null

return value	success true error false
Notes	

15.3.1 Set button event listener (setKeyListener)

Interface prototype	void setKeyListener(int keyType, IEventListener listener);	
Interface function	Set button event listener	
Parameter Description	Input Parameter	keyType - the type of button 1 : Help button Listener - button event callback, which can be null which indicating clear callback @see IEventListener
	Output Parameter	Null
return value	Null	

Notes	IEventListener callback event
	/**
	* button click event
	*
	* @param keyType button type
	* <pre>
	* 1 : Help button
	* </pre>
	*/
	Void onClick(int keyType);
	
	/**
	* The button is pressed for more than 2 seconds to trigger the event, triggering the event without triggering the button click event
	*
	* @param keyType button type
	* <pre>
	* 1 : Help button
	* </pre>
	*/
	Void onLongClick(int keyType);	

15、 Scale module statement

15.1 Scale Module statement

used for managing device scale module.

15.2 Package path

com.histonepos.npsdk.scale

15.3 AIDL file name

IScaleService.aidl

15.4 Interface list

15.4.1 Open scale(open)

Interface prototype	boolean open(String portSetting);	
Interface function	Open scale	
Parameter Description	Input Parameter	portSetting Port Setting
	Output Parameter	NULL
return value	True success,false fail	
Notes		

15.4.2 Close scale(close)

Interface prototype	Boolean close();	
Interface function	Close scale	
Parameter Description	Input Parameter	NULL
	Output Parameter	NULL
return value	True success,false fail	
Notes		

15.4.3 Get weight(getWeight)

Interface prototype	void getWeight(IScaleEventListener scaleListener);
Interface	Get Weight

function		
Parameter Description	Input Parameter	scaleListener event listener
	Output Parameter	NULL
return value	NULL	
Notes		

15.4.4 Set zero weight(setZeroWeight)

Interface prototype	int setZeroWeight();	
Interface function	Set Weight Zero	
Parameter Description	Input Parameter	NULL
	Output Parameter	NULL
return value	0-Sucess Other-Fail	
Notes		

15.4.5 Tare weight Physically(setTareWeight)

Interface prototype	int setTareWeight();	
Interface function		
Parameter Description	Input Parameter	NULL
	Output Parameter	NULL
return value	0-Sucess Other-Fail	
Notes	This method and “setDigitalTareWeight(int)”method are mutually exclusive.Only one of these two is available at the same time. Need perform exitTareWeight() first to change to another tare way.	

15.4.6 Tare weight Digitally(setTareWeight)

Interface prototype	int setDigitalTareWeight(int tare);

Interface function		
Parameter Description	Input Parameter	Tare ： tare weight to set
	Output Parameter	NULL
return value	0-Sucess Other-Fail	
Notes	This method and “setTareWeight()”method are mutually exclusive.Only one of these two is available at the same time. Need perform exitTareWeight() first to change to another tare way. 1. Use “setTareWeight()”instead of this method except special scenario.	

15.4.1 Config scale(config)

Interface prototype	Void config(Bundle type);	
Interface function	Config various scale	
Parameter Description	Input Parameter	Bundle bundle with scale type information。See demo for more
	Output Parameter	null
return value		
Notes		

15.5 Event Listener

```
interface IScaleEventListener{

    /**Data Change Event
     * @param strData    raw data
     */
     void    dataMessage(String strData);

    /**
     * weight change event
     * @param status
  ST: stabilization
UT:unstabilization
FT:offline
```

```
other:unkown error
        * @param weightType NW, GW
        * @param zeroPoint    zero point
        * @param weightVaule weight
        * @param unit    kg or g
        */
        void    weightChange(String status, String weightType,String zeroPoint,String weightVaule,
String unit);
}

/**
        *
        * @param status
ST:stable
UT:unstable
Flow:overflow or Gross weight less than -20d without tare subtraction
FT:offline    other:err
        * @param netWeight
        * @param tareWeight          minus: unsupport tare weight
        * @param unit    kg or g
        */
           void    onWeightBack(String status, float netWeight, float tareWeight,String unit);
```