Android SDK Interface Specification Document V4.0.2

Document Modification Record

Serial Number	Version Number	Modified Content	Modifier	Modification Date
1	v3.2.2	Creating an English version of SDK document	Bin Zhang	2024/3/21
2	V4.0.0	1. Function Optimization: 1. Update the image library, support QR code logo, adjustment of error correction level, and support adjustment of text word segmentation mode (English)	Bin Zhang	2025/8/12

		2. Optimize the Demo, modify the parameter passing method to call JSON data for printing, and prepare for the subsequent unified and rapid generation of JSON printing templates across various platforms 3. Optimize the Demo so that drawing and printing no longer need to concern themselves with the timing of data submission, which is instead automatically controlled by the encapsulated function, simplifying the calling logic. 4. SDK refactoring to enhance SDK stability		
3	V4.0.2	Function Optimization:	Bin Zhang	2025/10/27

		1. Update the image library to address the issue that the image library does not support 16KB pages on 64-bit devices 2. Optimize the Demo, with print data supporting JSON construction and object construction, and adapt it to the print data generated by the template generation tool. 3. Remove the remote dependency on the SpinKit Loading library in the Demo and change it to a local dependency		

Product Purpose

JCPrintApi Interface Method Specification Document, which provides explanations for the interface methods proposed for label drawing, facilitates users to call the interface during secondary development, shortens the development cycle, and accelerates the development progress.

Product Features

The JCPrintApi interface provides easy-to-use methods for callers to complete label drawing operations. This interface offers drawing methods for various objects such as text, one-dimensional barcodes, QR codes, images, and various graphics. It can also perform rotation of drawing objects, and callers can also call methods to obtain the completed label image for label preview, making label drawing operations more convenient.

Printer Support

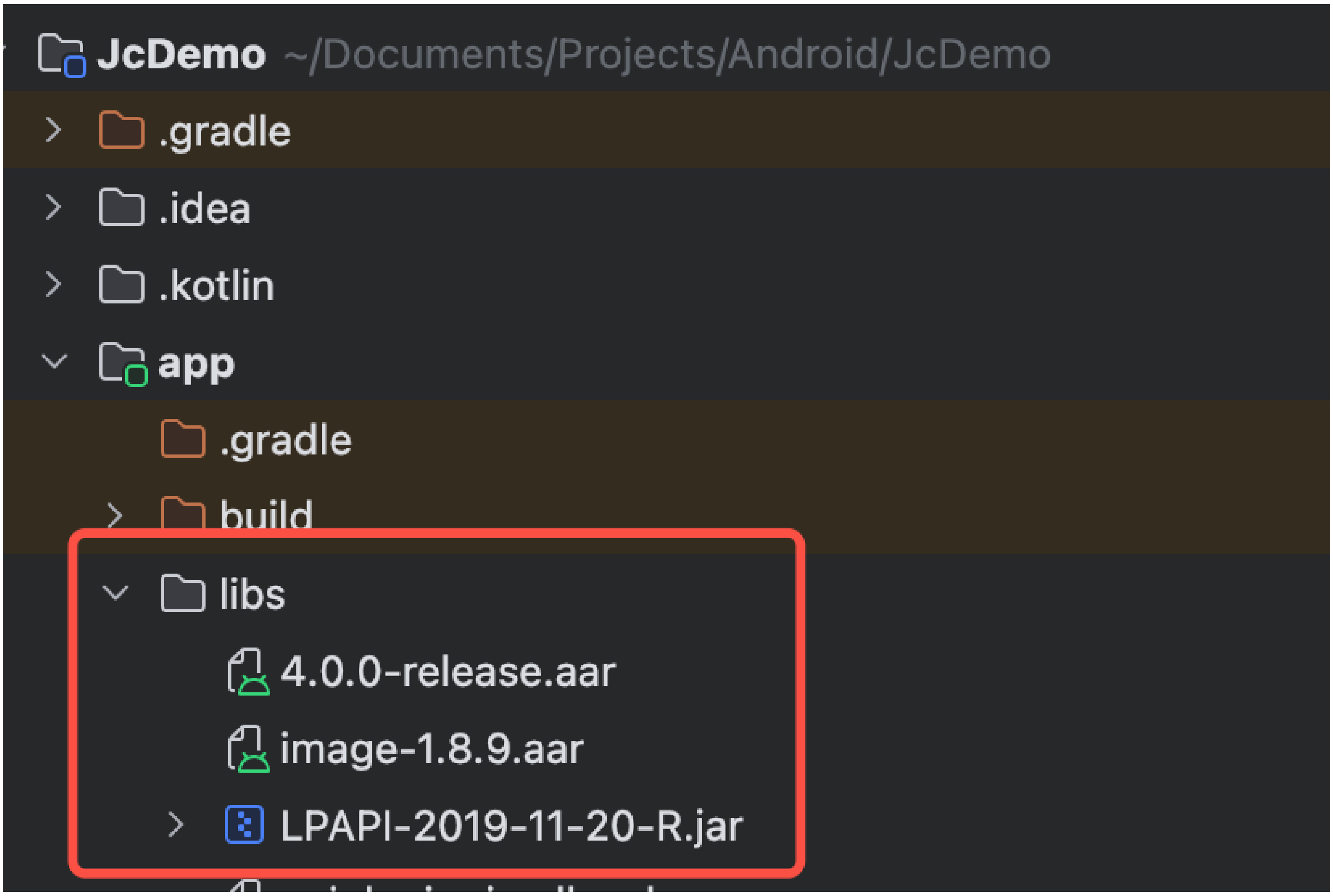
Supported Printer Models
H1
H1S
D11
D110
D101
B16
D110_M
D11_H
B21
B21S
B21_Pro
B11
B1
B203
B3S
B3S_P
B31
B4
N1(B18)

B50&B50W
B32&B32R&Z401
M2_H
M3
K2
K3
K3_W

1. Brief Introduction to the Use of JCPrintApi Interface

1.1 Import the SDK Library

1.1.1 Copy the SDK files to the lib folder



```
JcDemo ~/Documents/Projects/Android/JcDemo
  > .gradle
  > .idea
  > .kotlin
  v app
      .gradle
    > build
    v libs
        4.0.0-release.aar
        image-1.8.9.aar
      > LPAPI-2019-11-20-R.jar
```

1.1.2 Configuration of AAR Import in Gradle File

1.1.2.2 AAR below API31

代码块

```
android {
    // Minimum Android SDK version requirement: 19 or higher
    defaultConfig {
        applicationId "com.niimbot.jcdemo"
        minSdk 19
        targetSdk 29
        versionCode 1
        versionName "1.0"
        multiDexEnabled true
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }
    //JDK requires a minimum of 1.8.
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }

    repositories {
        flatDir{
            dir 'libs'
        }
    }

}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    // Print library
    implementation (name:'4.0.0-release',ext:'aar')
    implementation (name:'image-1.8.9',ext:'aar')
    // If the connected device models include printers from the B50/B50W/B11/T6/T7/T8 series, this package needs to be imported; if not, it can be skipped.
    implementation files('libs/LPAPI-2019-11-20-R.jar')
}
```

1.1.2.2 AAR introduced in API 31 and above

代码块

```
android {
    // Minimum Android SDK version requirement: 19 or higher
    defaultConfig {
        applicationId "com.niimbot.jcdemo"
        minSdk 19
        targetSdk 33
        versionCode 1
        versionName "1.0"
        multiDexEnabled true
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }
    //JDK requires a minimum of 1.8.
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }

}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    // Print library
    implementation files('libs/4.0.0-release.aar')
    implementation files('libs/image-1.8.9.aar')
    // If the connected device models include printers from the B50/B50W/B11/T6/T7/T8 series, this package needs to be imported; if not, it can be skipped.
    implementation files('libs/LPAPI-2019-11-20-R.jar')
}
```

1.2 Permissions required for using JCPrintApi

代码块

```

<!-- Bluetooth printer permissions -->
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />

<!-- Android 5 or above and Android 12 or below, Bluetooth search requires location permission. -->
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />

<!--Android 12 needs to apply for Bluetooth search permission, connection permission, but it doesn't need to apply for location permission -->
```

```
<uses-permission android:name="android.permission.BLUETOOTH_CONNECT" />
<uses-permission android:name="android.permission.BLUETOOTH_SCAN"
                 android:usesPermissionFlags="neverForLocation"
                 tools:targetApi="s" />


<!-- WIFI printer permission -->
<uses-permission android:name="android.permission.INTERNET"/>

<!-- WIFI search gives location permissions -->
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />

<!-- If you need to obtain the WIFI name of the current mobile phone connection, you need this permission. -->
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
```

1.3 Obfuscation Configuration

- Please avoid obfuscating the printing SDK, and add the following configuration to the Proguard obfuscation file:

代码块

```
-keep public class com.gengcon.www.jcprintersdk.**{*;}
-keep public class com.niimbot.canvas.image.**{*; }
-keep public class com.dothantech.**{*;}
-keep public class zpSDK.zpSDK.**{*;}
-keep public class com.snbc.sdk.**{*;}
-keep public class android_serialport_api.**{*;}

-dontwarn com.gengcon.www.jcprintersdk.**
-dontwarn com.jingchen.jcimagesdk.**
-dontwarn com.niimbot.canvas.image.**
-dontwarn com.dothantech.**
-dontwarn zpSDK.zpSDK.**
-dontwarn com.snbc.sdk.**
-dontwarn android_serialport_api.**
```

1.4 SDK Usage Requirements

1.4.1 System Version Requirements

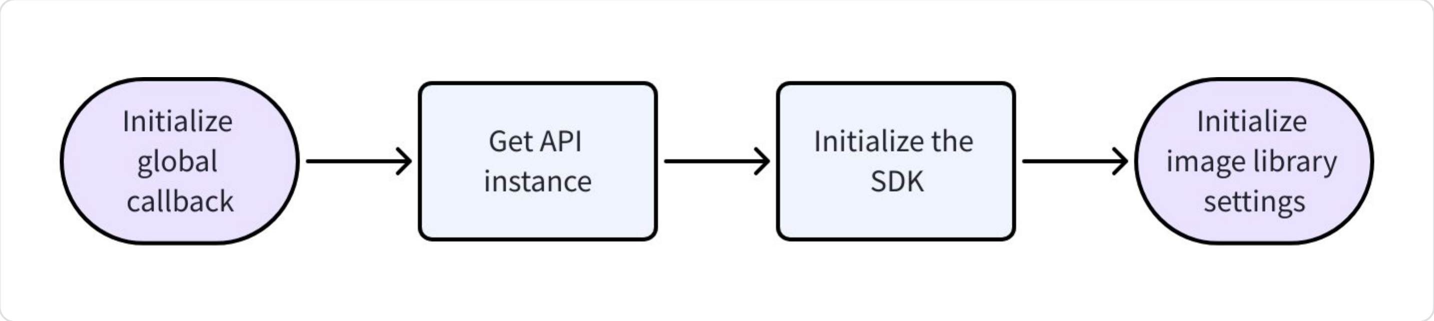
API 21 and above

1.4.2 CPU System Architecture

By default, it supports arm64-v8a or armeabi-v7a **(if there are special requirements, please contact technical support)**

2. Basic JCPrintApi Call Process

2.1 Interface Initialization Process



2.1.1 Printer Global Callback Interface Initialization

代码块

```
private static final Callback CALLBACK = new Callback() {

    /**
     * Successfully connection callback
     *
     * @param address Device address, Bluetooth is the Bluetooth MAC address,
     *                and WIFI is the IP address.
     *
     * @param type    Connection type, 0 means Bluetooth connection,
     *                1 means WIFI connection
     */
    @Override
    public void onConnectSuccess(String address, int type) {

    }


    /**
     * Disconnect callback.
     * This method is invoked when the device disconnects.
```

```java
     */
    @Override
    public void onDisConnect() {

    }


    /**
     * Charge change callback.
     * This method is called when the device's charge level changes.
     *
     * @param powerLevel The charge level, ranging from 1 to 4,
     *                   representing 1 to 4 bars of battery,
     *                    with 4 bars indicating a full charge.
     */
    @Override
    public void onElectricityChange(int powerLevel) {

    }


    /**
     * The status of lid change callback.
     * This method is invoked when the status of the lid changes. Currently,
     * this callback is supported only by the
     * printers H10/D101/D110/D11/B21/N1/B16/B32/Z401/B3S/B3S_P/B203/B1/B31/B4/N1/B18/K2/K3/M2/M3.
     *
     * @param coverStatus The status of the lid, where 0 means the cover is open
     *                     and 1 means the cover is closed.
     */
    @Override
    public void onCoverStatus(int coverStatus) {

    }


    /**
     * Monitor the status of the paper.
     * This method is called when the status of the paper is changed.
     * Currently, this callback is only supported
     * by the printers H10/D101/D110/D11/B21/B16/B32/Z401/B3S/B3S_P/B203/B1/B31/B4/N1/B18/K2/K3/M2/M3.
     *
     * @param paperStatus 0 means it is not out of paper, 1 means lack of paper.
     */
    @Override
    public void onPaperStatus(int paperStatus) {

    }

    /**
     * Monitor the RFID reading status of label paper.
     * This method is invoked when the RFID reading status of the label paper is changed.
     *
     * @param rfidReadStatus 0 indicates a failure to read the tag's RFID,
     *                       1 indicates a successful reading of the tag's RFID.
     *                       Currently, this callback is only supported by the
     *                       printers H10/D101/D110/D11/B21/B16/B32/Z401/B203/N1/
     *                       B18/K3/M2/M3.
     */
    @Override
    public void onRfidReadStatus(int rfidReadStatus) {

    }


    /**
     * Monitor the RFID reading status of the ribbon.
     * This method is called when the RFID read status of the ribbon is changed.
     *
     * @param ribbonRfidReadStatus 0 indicates a failure to read the ribbon's RFID,
     *                             1 indicates a successful reading of the
     *                             ribbon's RFID. Currently, this callback is
     *                             only supported by the printers N1/B18/B32/Z401/P1/      * P1S/M2/M3.
     */
    @Override
    public void onRibbonRfidReadStatus(int ribbonRfidReadStatus) {

    }

    /**
     * Monitor the status of the ribbon.
     * This method is invoked when the status of the ribbon is changed.
     *
     * @param ribbonStatus 0 indicates no ribbon at present,
     *                     1 indicates it has a ribbon at present.
     *                     Currently, this callback is supported only by
     *                     the printers N1/B18/B32/Z401/P1/P1S/M2/M3.
     */
    @Override
    public void onRibbonStatus(int ribbonStatus) {

    }

    /**
     * Firmware error callback, upgrade required.
     * This method is called when the device connects successfully
     * but detects abnormal firmware,
     * indicating it needs to do the firmware upgrade.
     */
    @Override
    public void onFirmErrors() {

    }
};
```

2.1.2 Initialize Printer Interface

代码块

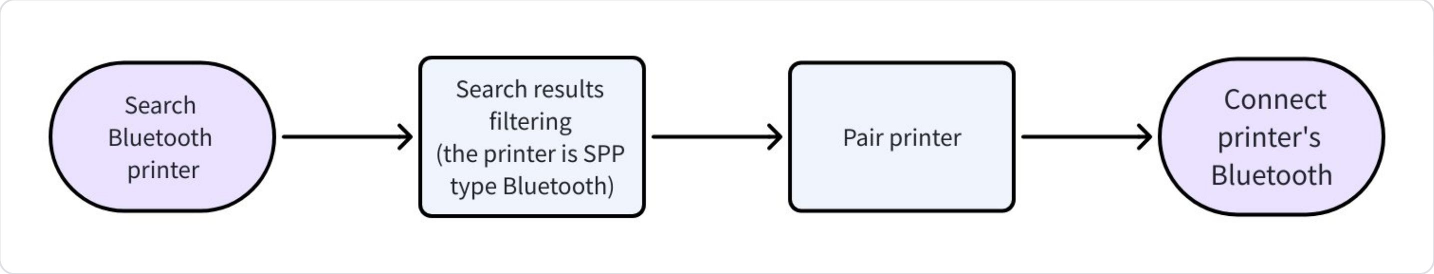
```
/**
* Retrieves the singleton instance of JCPrintApi.
*
* @return A JCPrintApi instance.
*/
public static JCPrintApi getInstance() {
    // Double-checked locking to ensure initialization only happens once
    if (api == null) {
        synchronized (PrintUtil.class) {
            if (api == null) {
                // Obtain the printer API instance
                api = JCPrintApi.getInstance(CALLBACK);
                // init has been deprecated, use initSdk replace,
                // which has a more accurate meaning, if you want to use the custom Application, please register in the manifest.
                api.initSdk(MyApplication.getInstance());
                // initImageProcessingDefault is deprecated,
                // replace with initDefaultImageLibrarySettings,
                // which has a more accurate meaning
                api.initDefaultImageLibrarySettings("", "");
            }
        }
    }

    return api;
}
```

2.2 Printer Connection Process

2.2.1 Bluetooth Printer Connection Process

Precautions:

- Please keep the Bluetooth and GPS of your phone or PDA turned on before searching for a printer
- App declares Bluetooth permissions
  - Android 12 and above declare Bluetooth search permission and Bluetooth connection permission
  - Declare Location Permission for Android 12 and Below
- App has been granted permissions
  - Android 12 and above Bluetooth search permission, Bluetooth connection permission
  - Location Permissions for Android 12 and Below


Search Bluetooth printer
Search results filtering (the printer is SPP type Bluetooth)
Pair printer
Connect printer's Bluetooth

2.2.2.1 Connect to the printer via Bluetooth

代码块

```
//This link is a synchronization time-consuming method,
//please use a child thread call.
api.connectBluetoothPrinter(blueDeviceInfo.getDeviceHardwareAddress());
```

2.2.2 WIFI Printer Connection Process

Precautions:

- Network configuration needs to be performed after connecting to the printer (Android supports configuring the printer network via Bluetooth, and it is also acceptable if you configure the Wi-Fi after connecting via USB on a PC)
- If the network configuration has already been completed, there is no need to perform these operations
- WIFI currently only supports 2.4G networks
- Currently, only the K3_W model printer supports network configuration
- Searching for WIFI printing requires declaring network permissions

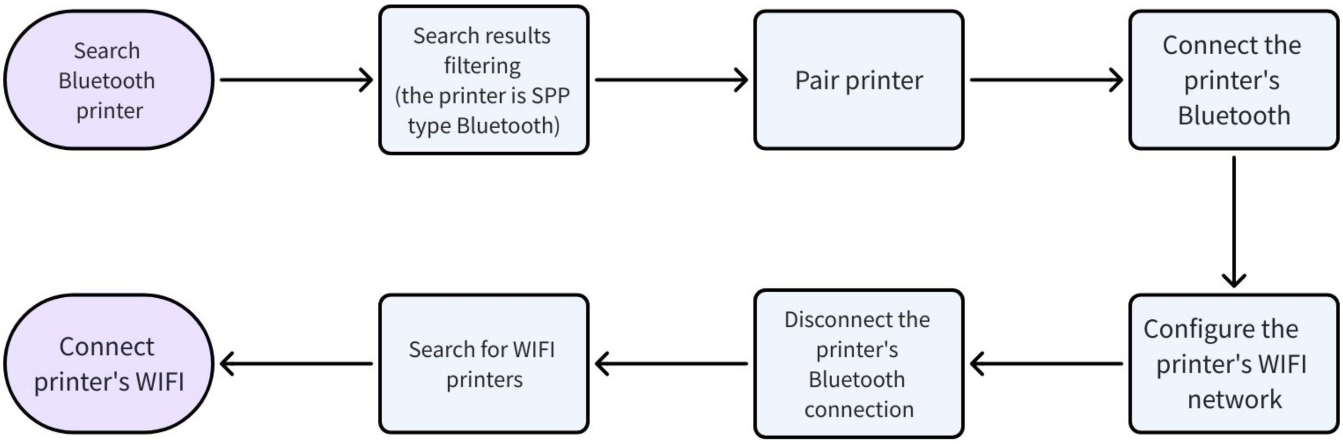
代码块

```
<uses-permission android:name="android.permission.INTERNET"/>
```

- To obtain the name of the currently connected Wi-Fi, you need to configure the permission for network state changes.

代码块

```
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
```


Search Bluetooth printer
Search results filtering (the printer is SPP type Bluetooth)
Pair printer
Connect the printer's Bluetooth
Configure the printer's WIFI network
Disconnect the printer's Bluetooth connection
Search for WIFI printers
Connect printer's WIFI

2.2.2.1 Configure Printer Network

代码块

```
/**
 * Configure the printer's network
 * (Call after a successful Bluetooth connection.
 * disconnect the printer's Bluetooth at first
 * then search for the printer's WiFi network.
 * Once found, you can connect to the printer.
 * This method is used to configure the printer's network,
 * including the name and password of the wireless network.
 *
 * @param wifiName     The name of the wireless network that the printer will
 *                     connect to.
 * @param wifiPassWord The password for the wireless network the printer will
 *                     connect to.
 * @return Configuration result:
 * 0: Configure successfully
 * -1: Configure failed
 * -3: Configuration not supported
 */
api.configurationWifi(wifiName,wifiPassWord)
```

2.2.2.2 Search for nearby WIFI printers

代码块

```
/**
 * Scans for nearby WiFi printers.
 * This method is used to search for WiFi printers in the nearby
 * and invokes the specified callback function after searching.
 * This method requires network permissions and location permissions.
 *
 * @param scanCallback The callback function is to handle the scanning
 *                     results.
 */
api.scanWifiPrinter(ScanCallback scanCallback)


/**
 * LAN search callback interface.
 */
public interface ScanCallback {
    /**
     * Callback Scan result.
     * This method is invoked when a device is detected,
     * then pass the device information.
     *
     * @param device Information about the scanned device.
     */
    void onScan(PrinterDevice device);
    /**
```

```
     * Scan completion callback.
     * This method is invoked when the scanning process is finished.
     */
    void onFinish();
}


/**
 * Search for device objects returned by the printer within the lan.
 */
public class PrinterDevice implements Serializable {
    /**
     * The name of device.
     */
    private String deviceName;
    /**
     * Device MAC address (Bluetooth MAC address).
     */
    private String deviceAddress;
    /**
     * IP address of the device.
     */
    private String deviceIp;
    /**
     * IP port number of the device.
     */
    public int port;
    /**
     * Number of connections.
     */
    public int connNum;
}
```

2.2.2.3 Connect wirelessly via the printer's IP Address and Port Number (asynchronous call required)

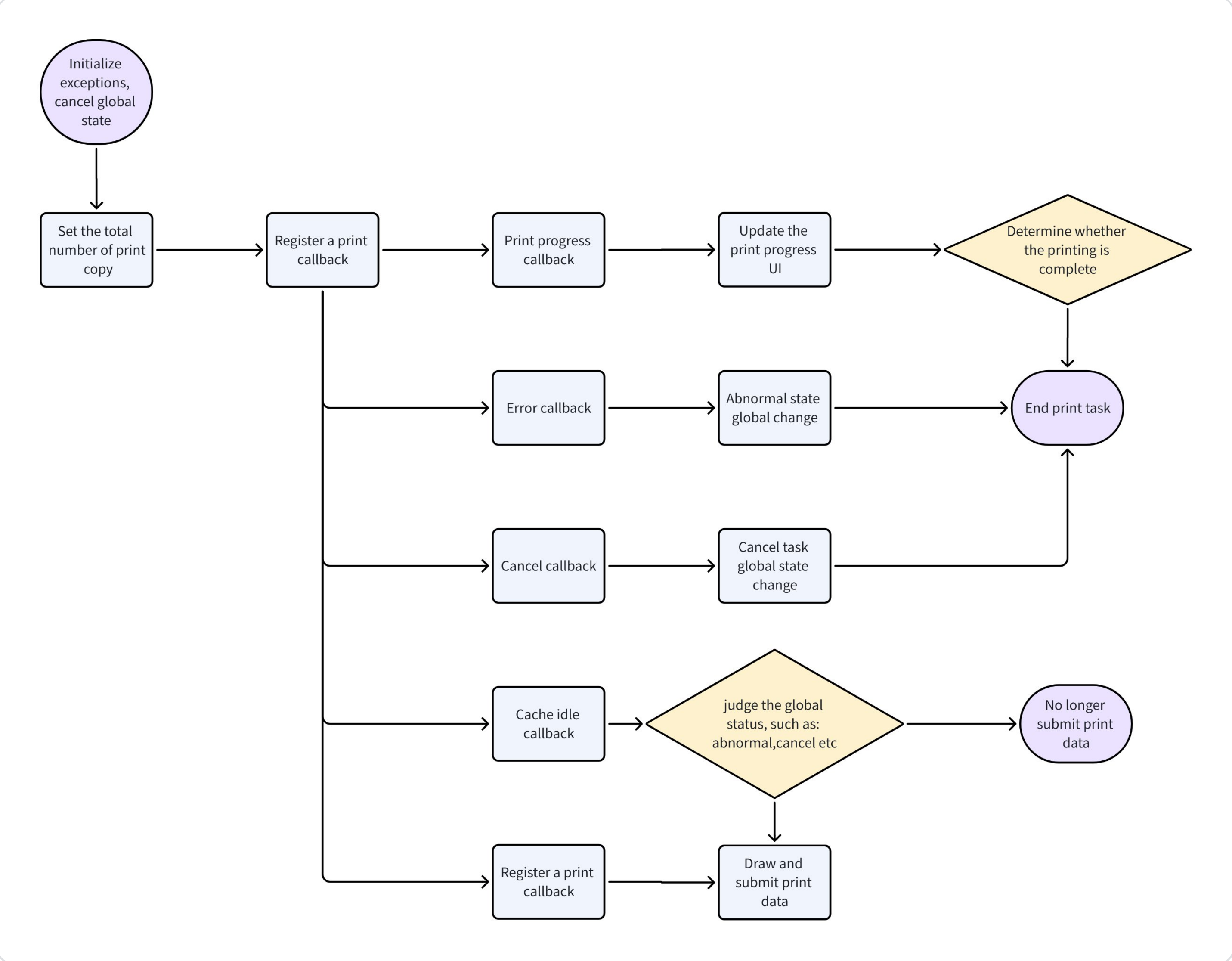
代码块

```
/**
 * Connects to a wireless printer.
 * This method is used to connect to prescribed wireless printers.
 * through pass hostname and IP address to connect it.
 *
 * @param ip IP address of the printer.
 * @param port port number of the printer.
 * @return Returns a connection result:
 * -3: doesn't support
 * -2: Connect busily
 * -1: Connect failed
 * 0: Connect successfully
 */
api.connectWifiPrinter(String ip, int port)
```

2.3 Printing Process

Precautions:

- After the print progress callback determines that the printing is completed, the print job needs to be ended via endJob; however, if the print job is terminated due to an exception or cancellation, there is no need to call endJob to end the print job
- **The parameter for setting the total number of print copies needs to match the actual number of copies to be printed (the sum of the number of print copies for all pages. For example, if you have 3 pages to print, with 3 copies for the first page, 2 copies for the second page, and 5 copies for the third page, then the value of count should be 10 (3+2+5).)**
- Print data submission must be done in the cache idle callback, and submission control is required. Print tasks that are abnormal, canceled, or have completed data submission must not be resubmitted to avoid errors (this callback will be reported periodically as long as there is cache idle while the print task is not completed). Additionally, the data submitted in a single operation must not exceed the cache space size


Initialize exceptions, cancel global state
Set the total number of print copy
Register a print callback
Print progress callback
Update the print progress UI
Determine whether the printing is complete
Error callback
Abnormal state global change
End print task
Cancel callback
Cancel task global state change
Cache idle callback
judge the global status, such as: abnormal,cancel etc
No longer submit print data
Register a print callback
Draw and submit print data

2.3.1 Register Callback

代码块

```java
PrintCallback printCallback = new PrintCallback() {
    /**
     * Print progress callback.
     * This method is invoked to provide progress information during printing,
     * including the number of printed pages,
     * the number of printed copies on the current page
     * and custom data (on models support RFID).
     *
     * @param pageIndex     The number of printed pages.
     * @param quantityIndex The number of printed copies on the current page.
     * @param customData    Custom data, used for models support RFID
     *                      callback data;
     *                      default handling is not necessary.
     */
    @Override
    public void onProgress(int pageIndex, int quantityIndex, HashMap<String, Object> hashMap) {
        runOnUiThread(() -> fragment.setStateStr("Print Progress: Completed page " + pageIndex + ", copy " + quantityIndex));
        // Print progress callback
        if (pageIndex == pageCount && quantityIndex == quantity) {
            //endJob, use the more explicit endPrintJob
            if (api.endPrintJob()) {
                Log.d(TAG, "Print ending succeeded");
            } else {
                Log.d(TAG, "Print ending failed");
            }


            handlePrintResult(fragment, "Print Successfully");
        }


    }

    @Override
    public void onError(int i) {

    }

    /**
     * Print error callback.
     * This method is invoked to report print errors during printing,
     * including error codes and the current print status.
     *
     * @param errorCode  Error code with specific meanings as follows:
     *                        1–The top cover of the printer is open;
     *                        2–Out of paper;
     *                        3–Low battery;
     *                        4–Battery is abnormal;
     *                        5–Manually stopped (button);
     *                        6–Data error;
     *                        7–Overheating;
     *                        8–Paper feed is abnormal;
     *                        9–Printing;
     *                        10–No printer head detected;
     *                        11–Temperature is too low;
     *                        12–Printer head unlocked;
     *                        13–Ribbon not detected;
     *                   14–Incorrect ribbon;
     *                   15–Ribbon depleted;
     *                   16–Unsupported paper type;
     *                   17–Paper type setting failed;
     *                   18–Print model setting failed;
     *                   19-Density setting failed;
     *                   20-RFID write failed;
     *                   21-Margin setting failed;
     *                   22-Communication abnormal;
     *                   23-Printer disconnect;
     *                   24-Canvas parameter error;
     *                   25-Rotation angle error;
     *                   26-JSON parameter error;
     *                   27-Paper feed abnormal (B3S);
     *                   28-Check paper type;
     *                   29-RFID tag not written;
     *                   30-Density setting not supported;
     *                   31-print model not supported;
     *                   32-Label material setting failed;
     *                   33-Unsupported label material setting.
     *                   34-RFID writing is not supported.
     *                   50-Illegal label.
     *                   51-Illegal ribbons and labels.
     *                   52-Firmware receive data timeout.
     *                   53-non-dedicated ribbon.
     *
     * @param printState Current print status,
     *                   where 0 indicates printing,
     *                   1 indicates paused,
     *                   and 2 indicates stopped.
     */
    @Override
    public void onError(int errorCode, int printState) {
        Log.d(TAG, "Test:onError");
        isError = true;
        String errorMsg = ERROR_MESSAGES.getOrDefault(errorCode, "");

        runOnUiThread(() -> {
            if (fragment != null) {
                fragment.dismiss();
            }
            Toast.makeText(MyApplication.getInstance(), errorMsg,
Toast.LENGTH_SHORT).show();
        });

    }

   /**
    * Cancel print task callback.
    * This method involves to callback cancel result when you cancel the print
task.
    *
    * @param isSuccess Indicates whether the cancellation was successful;
    *                  true for success, false for failure.
    */
    @Override
    public void onCancelJob(boolean isSuccess) {
        // Cancel print job callback successful
        isCancel = true;
    }

    /**
     * SDK cache queue idle callback.
     * This method is triggered when the SDK cache queue is idle,
     * indicating that print data can be submitted.
     *
     * @param pageIndex  Index of the next page to print.
     * @param bufferSize Size of the buffer.
     */
    @Override
    public void onBufferFree(int pageIndex, int bufferSize) {
        if (isError) {
            return;
        }
        if (pageIndex > pageCount) {
            return;
        }

        if (generatedPrintDataPageCount[0] < pageCount) {
            AssetManager assetManager = getResources().getAssets();
            Bitmap bitmap = null;
            try {
                InputStream is = assetManager.open("1.jpg");
                bitmap = BitmapFactory.decodeStream(is);
                is.close();

            } catch (Exception e) {
                e.printStackTrace();
            }


            float offsetX = 3.0F;
            float offsetY = 0.0F;
            Paint paint = new Paint();
            Canvas canvas = new Canvas();

            if (bitmap != null) {
                int bitmapWidth = bitmap.getWidth();
                int bitmapHeight = bitmap.getHeight();
                Bitmap backgroundBitmap = Bitmap.createBitmap(bitmapWidth, bitmapHeight, Bitmap.Config.ARGB_8888);
                canvas.setBitmap(backgroundBitmap);
                canvas.drawColor(Color.WHITE);
                canvas.drawBitmap(bitmap, offsetX * printMultiple, offsetY * printMultiple, paint);

                int commitDataLength = Math.min((pageCount - generatedPrintDataPageCount[0]), bufferSize);

                for (int i = 0; i < pageCount - generatedPrintDataPageCount[0]; i++) {
                    api.commitImageData(orientation, backgroundBitmap, (int) (bitmapWidth / printMultiple), (int) (bitmapHeight / printMultiple), 1, 0, 0, 0, 0, "");
                }
                generatedPrintDataPageCount[0] += commitDataLength;
            }


        }


    }
};
```

2.3.2 Set the total number of copies and start printing simultaneously

代码块

```
//Total number of pages
private int pageCount = 10;
//Number of print copy
private int quantity = 2;
//Set the total number of print copy
int totalQuantity = pageCount * quantity;
api.setTotalPrintQuantity(totalQuantity);
//To start printing, you need to pass in the concentration,
//paper type, print model, and print status callback.
api.startPrintJob(density, paperType, printMode, printCallback);
```

2.3.3 Generate JSON Print Data

代码块

```
private void generateMultiPagePrintData(int index, int cycleIndex) {
        while (index < cycleIndex) {
            float width = 70;
            float height = 50;
            int orientation = 0;
            float marginX = 2.0F;
            float marginY = 2.0F;
            //Rectangle type
            float rectangleWidth = width - marginX * 2;
            float rectangleHeight = height - marginY * 2;
            float lineWidth = 0.5F;
            //1. Circle 2. Ellipse 3. Rectangle 4. Rounded rectangle
            int graphType = 3;

            float lineHeight = rectangleHeight / 5.0F;

            float titleWidth = rectangleWidth * 2 / 5.0F;
            float contentWidth = rectangleWidth * 3 / 5.0F;

            float fontSize = 3.0F;
            float offsetY = 0.0F;
            float offsetX = -1.75F;



            //Set canvas size
            api.drawEmptyLabel(width, height, orientation, "");
            //Draw graphics
            api.drawLabelGraph(marginX +offsetX, marginY +offsetY, rectangleWidth, rectangleHeight, graphType, 0, 0, lineWidth, 1, new float[]{0.7575f, 0.7575f});

            //Draw lines
            api.drawLabelLine(marginX + offsetX, marginY + lineHeight - lineWidth + offsetY, rectangleWidth, lineWidth, 0, 1, new float[]{});
            api.drawLabelLine(marginX + offsetX, marginY + lineHeight * 2 - lineWidth + offsetY, rectangleWidth, lineWidth, 0, 1, new float[]{});
            api.drawLabelLine(marginX + offsetX, marginY + lineHeight * 3 - lineWidth + offsetY, rectangleWidth, lineWidth, 0, 1, new float[]{});
            api.drawLabelLine(marginX + offsetX, marginY + lineHeight * 4 - lineWidth + offsetY, rectangleWidth, lineWidth, 0, 1, new float[]{});

            api.drawLabelLine(marginX + titleWidth - lineWidth + offsetX, marginY + lineHeight + offsetY, lineWidth, rectangleHeight - lineHeight, 0, 1, new float[]{});

            //Use 6 in line feed model, the width and height are fixed, and the content is scaled when it is too large (the difference from model 1 is
that when the text content is typeset according to the size of budget font, and the width of the budget text box does not exceed the preset height after typesetting, the text will not be enlarged, but the text will be aligned to the text box according to the preset alignment method)
            api.drawLabelText(marginX * 3 + offsetX, marginY + offsetY, rectangleWidth - marginX * 4, lineHeight, "Wuhan Niimbot Intelligent Signage Technology Co., Ltd.", "", fontSize * 1.5F, 0, 1, 1, 6, 0, 1, new boolean[]{false, false, false, false});

            //Draw a subtitle
            api.drawLabelText(marginX * 2.5f + offsetX, marginY + lineHeight - lineWidth + offsetY, titleWidth - marginX * 3, lineHeight, "Asset name", "", fontSize, 0, 1, 1, 6, 0, 1, new boolean[]{false, false, false, false});
            api.drawLabelText(marginX * 2.5f + offsetX, marginY + lineHeight * 2 - lineWidth + offsetY, titleWidth - marginX * 3, lineHeight, "Asset number", "", fontSize, 0, 1, 1, 6, 0, 1, new boolean[]{false, false, false, false});
            api.drawLabelText(marginX * 2.5f + offsetX, marginY + lineHeight * 3 - lineWidth + offsetY, titleWidth - marginX * 3, lineHeight, "Enable date", "", fontSize, 0, 1, 1, 6, 0, 1, new boolean[]{false, false, false, false});
            api.drawLabelText(marginX * 2.5f + offsetX, marginY + lineHeight * 4 - lineWidth + offsetY, titleWidth - marginX * 3, lineHeight, "Storage location", "", fontSize, 0, 1, 1, 6, 0, 1, new boolean[]{false, false, false, false});

            api.drawLabelText(marginX * 2.5f + titleWidth + offsetX, marginY + lineHeight - lineWidth + offsetY, contentWidth - marginX * 3, lineHeight, "DELL monitor E6540", "", fontSize, 0, 0, 1, 6, 0, 1, new boolean[]{false, false, false, false});
            api.drawLabelText(marginX * 2.5f + titleWidth + offsetX, marginY + lineHeight * 2 - lineWidth + offsetY, contentWidth - marginX * 3, lineHeight, "C212004", "", fontSize, 0, 0, 1, 6, 0, 1, new boolean[]{false, false, false, false});
            api.drawLabelText(marginX * 2.5f + titleWidth + offsetX, marginY + lineHeight * 3 - lineWidth + offsetY, contentWidth - marginX * 3, lineHeight, "2014-06-10", "", fontSize, 0, 0, 1, 6, 0, 1, new boolean[]{false, false, false, false});
            api.drawLabelText(marginX * 2.5f + titleWidth + offsetX, marginY + lineHeight * 4 - lineWidth + offsetY, contentWidth - marginX * 3, lineHeight, (index+1)+"Office", "", fontSize, 0, 0, 1, 6, 0, 1, new boolean[]{false, false, false, false});

            //Generate print data
            byte[] jsonByte = api.generateLabelJson();

            //Convert to jsonStr
            String jsonStr = new String(jsonByte);


            jsonList.add(jsonStr);

            String jsonInfo = "{  " +
                    "\"printerImageProcessingInfo\": " + "{    " +
                    "\"orientation\":" + orientation + "," +
                    "   \"margin\": [      0,      0,      0,      0    ], " +
                    "   \"printQuantity\": " + quantity + ",  " +
                    "  \"horizontalOffset\": 0,  " +
                    "  \"verticalOffset\": 0,  " +
                    "  \"width\":" + width + "," +
                    "   \"height\":" + height + "," +
                    "  \"epc\": \"\"  }}";
            infoList.add(jsonInfo);
            index++;
        }
    }
```

2.3.4 Submit print data in the cache callback

2.3.4.1 Submit Print Data in JSON Format

代码块

```
@Override
public void onBufferFree(int pageIndex, int bufferSize) {
    // If an error occurs, printing has been cancelled, or the pageIndex exceeds the total number of pages, return
    if (isError || isCancel || pageIndex > pageCount) {
        return;
    }
    //The pages number of generated data is less than the total page number so that could generate data
    if (generatedPrintDataPageCount < pageCount) {
            int commitDataLength = Math.min((pageCount - generatedPrintDataPageCount), bufferSize);
        //The length of the data to be generated is less than the cacheable length
        // Generate data
        generateMultiPagePrintData(generatedPrintDataPageCount, generatedPrintDataPageCount + commitDataLength);
        // Submit print data
        api.commitData(jsonList.subList(generatedPrintDataPageCount, generatedPrintDataPageCount + commitDataLength), infoList.subList(generatedPrintDataPageCount, generatedPrintDataPageCount + commitDataLength));
        // Update the page number of generated print data
        generatedPrintDataPageCount += commitDataLength;

    }
}
```

2.3.4.2 Submit print data in bitmap mode

代码块

```
@Override
public void onBufferFree(int pageIndex, int bufferSize) {
    // If an error occurs, printing has been cancelled, or the pageIndex exceeds the total number of pages, return
    if (isError || isCancel || pageIndex > datasBitmap.size()) {
        return;
    }

    //Submit data
    api.commitImageData(orientation, datasBitmap.get(pageIndex-1), width, height, quantity, marginTop, marginLeft, marginBottoom, marginRight);
}
```

2.3.5 End printing after monitoring that printing is completed

代码块

```

    //After the printing is completed, call to end the printing to make the printer exit the printing state
    @Override
    public void onProgress(int pageIndex, int quantityIndex, HashMap<String, Object> hashMap) {

        runOnUiThread(() -> fragment.setStateStr("Print Progress: Completed page " + pageIndex + ", copy " + quantityIndex));
        Log.d(TAG, "Test: Print Progress: Completed up to page: " + pageIndex);
        // Print progress callback
        if (pageIndex == pageCount && quantityIndex == quantity) {
            Log.d(TAG, "Test:onProgress: Printing ended");
            //endJob, use the more explicit endPrintJob
            if (api.endPrintJob()) {
                Log.d(TAG, "Print ending succeeded");
            } else {
                Log.d(TAG, "Print ending failed");
            }


            handlePrintResult(fragment, "Print Successfully ");
        }


    }
```

3. Create a JCPrintApi Object

3.1 Create a JCPrintApi object

代码块

```
/**
 * Singleton access to ensure a unique interface instance.
 *
 * @param callBack The callback reference
 *                  (refer to the callback documentation).
 * @return The singleton instance of the interface.
 */
public static JCPrintApi getInstance(Callback callBack)
```

4. SDK Information Query

4.1 Get SDK Version Number

代码块

```
/**
 * Retrieves the SDK version number.
 * This method is used to obtain the version number of the current SDK,
 * which follows the format "major.minor.patch-release".
 *
 * @return The version number of the current SDK will return with a string style,
 * for example, "3.1.9-release".
 */
public String getSdkVersion()
```

4.2 Obtain the label size installed in the printer

代码块

```
/**
 * Get the label size installed in the printer (currently only supports M2 models with firmware version V1.24 or higher)
 * Notes: When state is 0 and the paperType parameter is not 0, the read parameters are valid
 *
 * @return PaperInfo object, containing label type, gap height, paper width, paper height, tail direction (pigtail), and tail length
 */
public PaperInfo getPaperInfo()


public class PaperInfo {
    // -1 for failure, 0 for success
    public int state = -1;
    // Gap height (black mark height) (unit: pixel)
    public int gapHeightPixel;
    // Paper height (including gap) (unit: pixels)
    public int totalHeightPixel;
    // Paper Type: 1: Gap paper; 2: Black mark paper; 3: Continuous paper; 4: Perforated paper; 5: Transparent paper; 6: Label;10:Black Label Gap Paper
    public int paperType;
    //Gap height (black mark height) (unit: millimeter)
    public float gapHeight;
    //Paper height (including gap) (unit: mm)
    public float totalHeight;
    //Paper width (including gap) (unit: pixels)
    public int paperWidthPixel;
    //Paper width (including gap) (unit: mm)
    public float paperWidth;
    //Tail direction: 1 up, 2 down, 3 left, 4 right (temporarily not supported)
    public int direction;
    //Tail length (unit: pixel)
    public int tailLengthPixel;
    //Tail length (unit: millimeter)
    public float tailLength;

    public PaperInfo() {
    }

    public String toString() {
        return "PaperInfo{state=" + this.state + ", gapHeightPixel=" + this.gapHeightPixel + ", totalHeightPixel=" + this.totalHeightPixel + ", pagerType=" + this.paperType + ", gapHeight=" + this.gapHeight + ", totalHeight=" + this.totalHeight + ", pagerWidthPixel=" + this.paperWidthPixel + ", pagerWidth=" + this.paperWidth + ", direction=" + this.direction + ", tailLengthPixel=" + this.tailLengthPixel + ", tailLength=" + this.tailLength + '}';
    }
}
```

5. Printer Settings and Queries

5.1 Set the printer's automatic shutdown time

代码块

```java
/**
 * Sets the printer's automatic shutdown time.
 * This method is used to set the automatic shutdown timer of the printer.
 *
 * @param autoShutDownTimeLevel The level of automatic shutdown time,
 *                              with available options:
 *                              1: Automatically shuts down after 15 minutes
 *                              2: Automatically shuts down after 30 minutes
 *                              3: Automatically shuts down after 45 minutes
 *                              4: Never automatically shuts down
 *                              (D-series of printer defaults to 45 minutes)
 * @return The result of the setting operation:
 * 0: Setting successfully
 * -1: Setting failed
 * -2: Setting not supported
 */
public int setPrinterAutoShutdownTime(int autoShutDownTimeLevel)
```

5.2 Print Magnification Query

代码块

```java
/**
 * Get the print scale multiplier.
 * This method is used to retrieve the ratio for
 * converting millimeters to pixels,
 * which calculated based on the resolution of printer.
 *
 * @return The current print scale multiplier.
 */
public float getPrintScaleMultiplier()
```

6. Introduction to JCPrintApi.Callback Related Methods

6.1 Connection Success Callback

代码块

```
/**
 * Connection successful callback
 *
 * @param address Device address, Bluetooth is the Bluetooth MAC address,
 *                and WIFI is the IP address.
 *
 * @param type    Connection type, 0 means Bluetooth connection,
 *                1 means WIFI connection
 */
void onConnectSuccess(String deviceAddress, int connectType);
```

6.2 Connection Disconnect (Failure) Callback

代码块

```
/**
 * Disconnected callback.
 * This method is invoked when the device disconnects.
 */
void onDisConnect();
```

6.3 Callback for Lid Closed State Change

代码块

```
/**
 * Lid status change callback.
 * This method is invoked when the status of lid is changed. Currently,
 * this callback is supported only by the
 * printers H10/D101/D110/D11/B21/B16/B32/Z401/B3S/B3S_P/B203/B1/B31/B4/N1/B18/K2/K3/M2/M3.
 *
 * @param coverStatus The status of lid, where 0 indicates the cover is open
 *                    and 1 indicates the cover is closed.
 */
void onCoverStatus(int coverStatus);
```

6.4 Power Change Callback

代码块

```
/**
 * Electricity change callback.
 * This method is called when the device's charge level changes.
 * Currently, this callback is supported only by the
 * printers H10/D101/D110/D11/B21/B16/B32/Z401/B3S/B3S_P/B203/B1/B31/B4/N1/B18/M2/M3.
 *
 * @param powerLevel The charge level, ranging from 1 to 4,
 *                   representing 1 to 4 bars of battery,
 *                    with 4 bars indicating a full charge.
 */
void onElectricityChange(int powerLevel);
```

6.5 Printer Paper Status Change Callback

代码块

```
/**
 * Monitor the status of the paper.
 * This method is called when the status of the paper is changed.
 * Currently, this callback is only supported
 * by the printers H10/D101/D110/D11/B21/B16/B32/Z401/B3S/B3S_P/B203/B1/B31/B4/N1/B18/K2/K3/M2/M3.
 *
 * @param paperStatus 0 means it is not out of paper, 1 means lack of paper.
 */
void onPaperStatus(int paperStatus);
```

6.6 Callback for RFID Tag Reading Status Change

代码块

```
/**
 * Monitors the RFID tag read status changes.
 * This method is called whenever there's a change in the status
 * of reading an RFID tag.
 *
 * @param rfidReadStatus An integer representing the RFID read status:
 *                      - 0 indicates that no RFID tag has been read.
 *                      - 1 signifies a successful read of an RFID tag.
 *                      Currently, this callback is supported only for
 *
H10/D101/D110/D11/B21/B16/B203/B1/B31/B4/N1/B18/M2/K3/B3S_P
 *                       series printers.
 */
void onRfidReadStatus(int rfidReadStatus);
```

6.7 Ribbon Status Change Callback

代码块

```

/**
 * Monitor the status of the ribbon.
 * This method is invoked when the status of the ribbon is changed.
 *
 * @param ribbonStatus 0 indicates no ribbon at present,
 *                     1 indicates it has a ribbon at present.
 *                     Currently, this callback is supported only by
 *                     the printers N1/B18/B32/Z401/P1/P1S/M2/M3.
 */
void onRibbonStatus(int ribbonStatus);
```

6.8 Carbon Ribbon RFID Read Status Change Callback

代码块

```
/**
 * Monitor RFID ribbon read status changes.
 * This method is called when the RFID read status of the ribbon changes.
 *
 * @param ribbonRfidReadStatus 0 indicates failure to read the ribbon's RFID,
 *                             1 indicates successful reading of the
 *                             ribbon's RFID.Currently, this callback is
 *                             only supported by the printers
 *                             N1/B18/B32/Z401/P1/P1S/M2/M3
 */
void onRibbonRfidReadStatus(int ribbonRfidReadStatus);
```

6.9 Firmware Exception Callback

代码块

```
/**
 * 固件异常回调，需要升级
 * 当设备连接成功但出现固件异常时，将调用此方法，表示需要进行固件升级。
 */
void onFirmErrors();
```

VII. Instructions on Methods Related to Linking Printers

~~7.1 Turn on the printer via Bluetooth connection using the printer's MAC address (synchronous) (deprecated)~~

代码块

```
/**
 * Opens a Bluetooth connection to the printer
 * by its MAC address (synchronously).
 * Bluetooth connection permission is required When the API for engineering adaptation is greater than or equal to 31.
 *
 * @param address The Bluetooth MAC address of the printer.
 * @return Returns the connection result:
 * -3: Not supported
 * -2: Connection busy
 * -1: Connection failed
 * 0: Connection successful
 * @deprecated This method is deprecated;
 * please use the connectBluetoothPrinter method instead.
 */
@Deprecated
public int openPrinterByAddress(String address)
```

7.2 Turn on the printer (synchronously) by establishing a Bluetooth connection via the printer's MAC address

代码块

```
/**
 * Opens a Bluetooth connection to the printer
 * by its MAC address (synchronously).
 * Bluetooth connection permission is required for API levels 31 and above.
 *
 * @param address The Bluetooth MAC address of the printer.
 * @return Returns the connection result:
 * -3: Not supported
 * -2: Connection busy
 * -1: Connection failed
 * 0: Connection successful
 */
public int connectBluetoothPrinter(String address)
```

7.3 Configure Printer Network

代码块

```
/**
 * Configure the printer's network
 * (Call after a successful Bluetooth connection.
 * disconnect the printer's Bluetooth at first
 * then search for the printer's WiFi network.
 * Once found, you can connect to the printer.
 * This method is used to configure the printer's network,
 * including the name and password of the wireless network.
 *
 * @param wifiName     The name of the wireless network that the printer will
 *                     connect to.
 * @param wifiPassWord The password for the wireless network the printer will
 *                     connect to.
 * @return Configuration result:
 * 0: Configure successfully
 * -1: Configure failed
 * -3: Configuration not supported
 */
public int configurationWifi(String wifiName, String wifiPassWord)
```

7.4 Search for nearby WIFI printers

代码块

```
/**
 *  * Scans for nearby WiFi printers.
 * This method is used to search for WiFi printers in the nearby
 * and invokes the specified callback function after searching.
 * This method requires network permissions and location permissions.
 *
 * @param scanCallback The callback function is to handle the scanning
 *                     results.
 */
public void scanWifiPrinter(ScanCallback scanCallback)

/**
 * LAN search callback interface.
 */
public interface ScanCallback {
    /**
     * Callback Scan result.
     * This method is invoked when a device is detected,
     * then pass the device information.
     *
     * @param device Information about the scanned device.
     */
    void onScan(PrinterDevice device);
    /**
     * Scan completion callback.
     * This method is invoked when the scanning process is finished.
     */
    void onFinish();
}


/**
```

```java
 * Search for device objects returned by the printer within the lan.
 */
public class PrinterDevice implements Serializable {
    /**
     * Device name.
     */
    private String deviceName;
    /**
     * Device MAC address (Bluetooth MAC address).
     */
    private String deviceAddress;
    /**
     * Device IP address.
     */
    private String deviceIp;
    /**
     * IP port number of the device.
     */
    public int port;
    /**
     * Number of connections.
     */
    public int connNum;
}
```

7.5 Connect wirelessly via the printer's IP Address and Port Number (asynchronous call required)

代码块

```java
/**
 * Connects to a wireless printer.
 * This method is used to connect to prescribed wireless printers.
 * through pass hostname and IP address to connect it.
 *
 * @param ip IP address of the printer.
 * @param port port number of the printer.
 * @return Returns a connection result:
 * -3: doesn't support
 * -2: Connect busily
 * -1: Connect failed
 * 0: Connect successfully
 */
public int connectWifiPrinter(String ip, int port)
```

7.6 Obtain the configured wireless network name

代码块

```
/**
 * Get the name of configured wireless name.
 * This method is used to obtain the name
 * of the wireless network that has been configured.
 *
 * @return Returns the name of configured wireless network:
 * Success: Returns the name of actual wireless network
 * "-1": retrieve failed
 * "-3": doesn't support
 */
public String getConfigurationWifiName()
```

7.7 Disconnect the Printer

代码块

```
/**
 * Disconnect the printer.
 *
 * @description After calling,
 * a callback response will be triggered indicating the disconnection.
 *
 */
public void close()
```

8. Explanation of Printing-Related Methods

~~8.1 Set Total Print Copies (Deprecated)~~

代码块

```
/**
 * Sets the total number of print cpoy.
 * This method is used to set the total number of print copies to
 * show the sum of print copy within all pages.
 * For example, if you have 3 pages to print,
 * 3 copies on the first page,
 * 2 copies on the second page, and 5 copies on the third page,
 * then the value of count should be 10 (3 + 2 + 5).
 *
 * @param quantity The total number of print copy to be set.
 * @deprecated This method is deprecated;
 * please use the setTotalPrintQuantity method instead.
 */
@Deprecated
public void setTotalQuantityOfPrints(int quantity)
```

8.2 Set the total number of copies to print

代码块

```
/**
 * Sets the total number of print cpoy.
 * This method is used to set the total number of print copies to
 * show the sum of print copy within all pages.
 * For example, if you have 3 pages to print,
 * 3 copies on the first page,
 * 2 copies on the second page, and 5 copies on the third page,
 * then the value of count should be 10 (3 + 2 + 5)
 * @param quantity The total number of prints to be set.
 */
public void setTotalPrintQuantity(int quantity)
```

8.3 Start the printing task

代码块

```
/**
 * Begins a print task, including setting print density, paper type,
 * print model,and specifying a print status callback.
 *
 * Important:
 * Prior to invoking this method,
 * ensure `setTotalPrintQuantity` has been called and
 * the printer has been connected.
 *
 * @param density       The print density,
 *                      which varies depending on the different printer model:
 *                      D11, D101, D110, H10, B16, B18、N1: 1~3, default 2
 *                      B3S,B203,B1,B2O3,B31,B4,K3,K3W,K2,M2,M3: 1~5, default 3
 *                      B50, B11, B50W, B32, Z401: 1~15, default 8
 * @param paperType     The paper type, with available options:
 *                      1: Gap paper;
 *                      2: Black-marked paper;
 *                      3: Continuous paper;
 *                      4: Perforated paper;
 *                      5: Transparent paper; 6: Label
 *                      10:Black Label Gap Paper
 * @param printMode     The print model, with available options:
 *                      1: Thermal; 2: Thermal transfer
 *                      D11, D101, D110, H10, B16, B3S, B203, B1,B2O3,B31,B4,K2 K3, K3W, B11
 *                      only support thermal
 *                      B50, B50W, B32, Z401 , M2,M3,N1, B18
 *                      only support thermal transfer
 * @param printCallback The print status callback
 */
public void startPrintJob(int density, final int paperType, final int printMode, final PrintCallback printCallback)
```

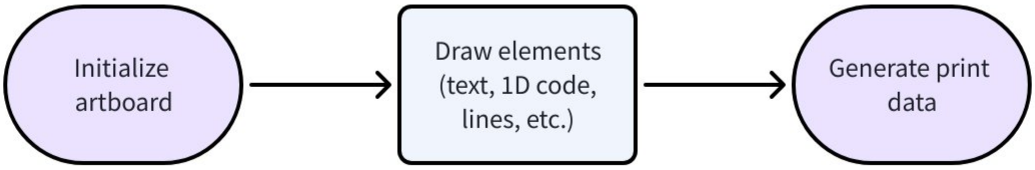
8.4 Submit Print Data

8.4.1 Submit data in the form of a Json list

代码块

```
/**
 * Submits printing data.
 * The submission should occur within the print callback `onBuffer`.
 *
 * @param printDataList   A list containing the print data.
 * @param printerInfoList A list of printer information JSON strings.
 *                        If an RFID is included (supported only in B32R models),
 *                        ensure that the printer info JSON contains
 *                        an 'epc' field.
 *                        The 'epc' value must be a string with a length
 *                        divisible by 4,
 *                        consisting of 0-9 and A-F characters
 *                        (case insensitive).
 *                        It's formatted as a sequence of pairs
 *                        representing hexadecimal values,
 *                        e.g., "012345678". Each pair corresponds to
 *                        a single 16-bit hexadecimal number,
 *                        like "0102" which would represent 0x01 and 0x02.
 */
public void commitData(final List<String> printDataList, final List<String> printerInfoList)
```

8.4.1.1 Flowchart for printData Generation



Sample code is as follows:

代码块

```
float width = 60;
float height = 40;
int orientation = 0;
float marginX = 2.0F;
float marginY = 2.0F;
float rectangleWidth = width - marginX * 2;
float rectangleHeight = height - marginY * 2;
float fontSize = 3.0F;
float offsetY = 0F;
float offsetX = 0F;
float lineWidth = 0.5F;
float lineHeight = rectangleHeight / 5.0F;
// Set the size of canvas
PrintUtil.getInstance().drawEmptyLabel(width, height, orientation, "");
// Draw text
PrintUtil.getInstance().drawLabelText(2.5F, 2.0F, rectangleWidth, rectangleHeight, "Model", "", fontSize, 0, 0, 0, 6, 0, 1, new boolean[]{false, false, false, false});
//Draw lines
PrintUtil.getInstance().drawLabelLine(marginX + offsetX, marginY + lineHeight - lineWidth + offsetY, rectangleWidth, lineWidth, 0, 1, new float[]{});
//Generate print data
byte[] jsonByte = PrintUtil.getInstance().generateLabelJson();
```

8.4.1.2 printInfo Format Description

```
//Z401/B32/B32R/T8/M2/M3/B21_pro are 300dpi models, other models are 200dpi
{
  "printerImageProcessingInfo": {//Print additional information
    "orientation": 0,//Rotation angle
    "margin": [//Margins
      0,
      0,
      0,
      0
    ],
    "printQuantity": 1,//Used to specify the number of print copies on the current page. For example, if you need to print 3 pages, 3 copies on the first page, 2 copies on the second page, and 5 copies on the third page, then the printQuantity value should be 3, 2, and 5 for each of the 3 commits.
    "horizontalOffset": 0,//horizontal offset
    "verticalOffset": 0,//vertical offset
    "width": 60,//Canvas width in mm
    "height": 40,//Canvas height in mm
    "printMultiple": 8,//Printers with a print magnification of 200dpi are 8, and printers with 300dpi are 11.81.
    "epc": ""//RFID tag writes data, B32R machine fills data when printing RFID tag, other scenarios default "
  }
}
```

8.4.2 Submit data in Bitmap format

代码块

```
/**
 * Submits image print data,
 * This mthod is used to submit print data and trigger a print task, include
 * print orientation, bitmap image,page dimensions, quantity,
 * margins, and RFID information.
 *
 * @param orientation   The print orientation, with values 0, 90, 180, or 270,
 *                      each representing rotation in degrees (0, 90, 180, 270).
 * @param printBitmap   The bitmap image to be printed.
 * @param pageWidth     The width of the print page in millimeters (mm).
 * @param pageHeight    The height of the print page in millimeters (mm).
 * @param printQuantity The number of print copy.
 * @param marginTop     The top margin,
 *                      defining the distance from the print content to
 *                      the top edge of the page in mm.
 * @param marginLeft    The left margin,
 *                      defining the distance from the print content to
 *                      the left edge of the page in mm.
 * @param marginBottom  The bottom margin,
 *                      defining the distance from the print content to
 *                      the bottom edge of the page in mm.
 * @param marginRight   The right margin,
 *                      defining the distance from the print content to
 *                      the right edge of the page in mm.
 * @param rfid          The RFID data to be written into the print data
 *                      (for RFID tag writing on B32R machines;
 *                      default is "" for other scenarios).
 *                      The data length must be a multiple of 4,
 *                      composed of characters 0-9 and A-F (case-insensitive),
 *                      formatted as:"012345678",
 *                      where each pair of characters represents
 *                      a single hexadecimal value
 *                      (e.g., "0102" corresponds to 0x01 and 0x02).
 */
public void commitImageData(final int orientation, final Bitmap printBitmap, final float pageWidth, final float pageHeight, final int printQuantity, final int marginTop, final int marginLeft, final int marginBottom, final int marginRight, final String rfid)
```

~~8.5 End Print Task (Obsolete)~~

代码块

```
/**
 * Ends print task.
 * This method is used to terminate the ongoing print task.
 * It's a time-consuming operation,
 * typically called at the end of the last page.
 * For the completeness of the printing flow,
 * this method must be invoked before
 * starting the next print task (startPrintTask()).
 *
 * @return Returns true if the print task termination is successful;
 * otherwise, returns false.
 * @deprecated This method is deprecated.
 * Please use the {@code endPrintJob} method instead.
 */
```

```
@Deprecated
public boolean endJob()
```

8.6 End Print Task

代码块

```
/**
 * Ends the current print job.
 * This method is used to terminate the ongoing print task.
 * It's a time-consuming operation,
 * typically called at the end of the last page.
 * For the completeness of the printing flow,
 * this method must be invoked before
 * starting the next print job (startPrintJob()).
 *
 * @return Returns true if the print job termination is successful;
 * otherwise, returns false.
 */
public boolean endPrintJob()
```

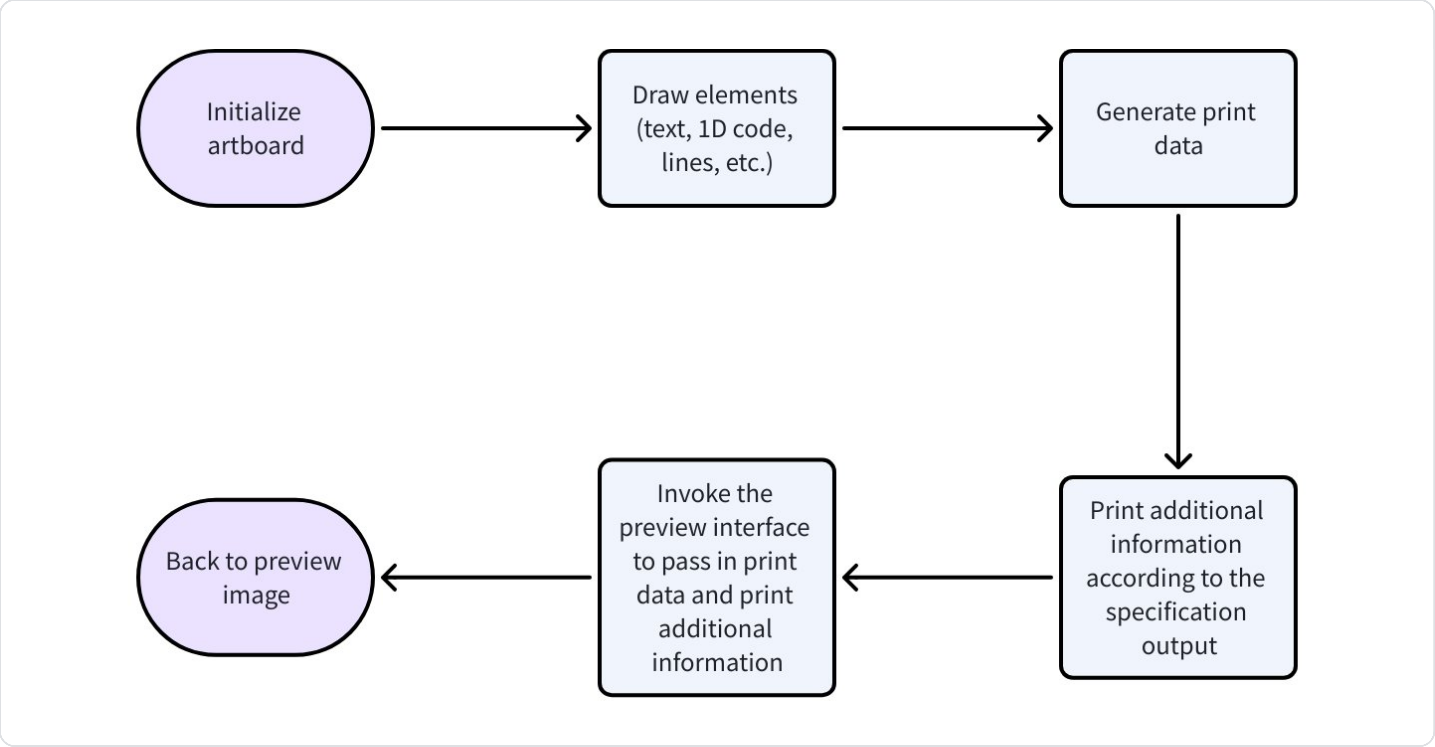
8.7 Cancel Print Task

代码块

```
/**
 * Cancels the current print job.
 * This method is used to abort the ongoing print task.
 * It's a time-consuming operation,
 * and the result of the cancellation can also
 * be determined through a cancellation callback.
 * @return Returns true if the print task cancellation is successful;
 * otherwise, returns false.
 */
public boolean cancelJob()
```

8.8 Print Preview

8.8.1 Preview Process


Initialize artboard
Draw elements (text, 1D code, lines, etc.)
Generate print data
Print additional information according to the specification output
Invoke the preview interface to pass in print data and print additional information
Back to preview image

8.8.2 Print Preview Interface Description

代码块

```
/**
 * Generates a print preview image.
 * This method is used to creat a preview image of the print content,
 * requiring the print data and additional print information.
 *
 * @param printData   The specific printing content.
 * @param printerInfo Print inadditional information
 *                    including printer details, etc.
 * @return A Bitmap object representing the preview image.
 */
public Bitmap generatePreviewImage(String printData, String printerInfo)
```

IX. Introduction to Printer Process Callback-Related Methods

9.1 Print Progress Callback

代码块

```
/**
 * Print Progress Callback
 * This method is used to callback print progress on the print progress,
 * including the page number that has been printed,
 * the completed print copy for the current page,
 * and custom data (specifically for models supporting RFID functionality).
 *
 * @param pageIndex The number of pages that have been printed so far.
 * @param quantityIndex The completed print copy for the current page.
 * @param customData Custom data,
 *                   used for providing the result of
 *                   RFID tag writing during printing in devices that support
 *                   RFID,default to be ignored if not applicable.
 */
void onProgress(int pageIndex, int quantityIndex, HashMap<String, Object>
customData);
```

9.2 Print Exception Callback

代码块

```
/**
 * Print error callback.
 * This method is invoked to report print errors during printing,
 * including error codes and the current print status.
 *
 * @param errorCode  Error code with specific meanings as follows:
 *                        1–The top cover of the printer is open;
 *                        2–Out of paper;
 *                        3–Low battery;
 *                        4–Battery is abnormal;
 *                        5–Manually stopped (button);
 *                        6–Data error;
 *                        7–Overheating;
 *                        8–Paper feed is abnormal;
 *                        9–Printing;
 *                        10–No printer head detected;
 *                        11–Temperature is too low;
 *                        12–Printer head unlocked;
 *                        13–Ribbon not detected;
 *                   14–Incorrect ribbon;
 *                   15–Ribbon depleted;
 *                   16–Unsupported paper type;
 *                   17–Paper type setting failed;
 *                   18–Print model setting failed;
 *                   19–Density setting failed;
 *                   20–RFID write failed;
 *                   21–Margin setting failed;
 *                   22–Communication abnormal;
 *                   23–Printer disconnect;
 *                   24–Canvas parameter error;
 *                   25–Rotation angle error;
 *                   26–JSON parameter error;
 *                   27–Paper feed abnormal (B3S);
 *                   28–Check paper type;
 *                   29–RFID tag not written;
 *                   30–Density setting not supported;
 *                   31–print model not supported;
 *                   32–Label material setting failed;
 *                   33–Unsupported label material setting.
 *                   34–RFID writing is not supported.
 *                   50–Illegal label.
 *                   51–Illegal ribbons and labels.
 *                   52–Firmware receive data timeout.
 *                   53–non-dedicated ribbon.
 *
 * @param printState Current print status,
 *                   where 0 indicates printing,
 *                   1 indicates paused,
 *                   and 2 indicates stopped.
 */
void onError(int errorCode, int printState);
```

9.3 Cancel the print completion callback

代码块

```
/**
 * Cancel print task callback.
 * This method involves to callback cancel result when you cancel the print task.
 *
 * @param isSuccess Indicates whether the cancellation was successful;
 *                  true for success, false for failure.
 */
void onCancelJob(boolean isSuccess);
```

9.4 SDK Remaining Cache Callback

代码块

```
/**
 * SDK cache queue idle callback.
 * This method is triggered when the SDK cache queue is idle,
 * indicating that print data can be submitted.
 *
 * @param pageIndex  Index of the next page to print.
 * @param bufferSize Size of the buffer.
 */
void onBufferFree(int pageIndex, int bufferSize);
```

10. JSON Drawing Interface Call

~~10.1 Initialize SDK (deprecated)~~

代码块

```
/**
 * Initializes the SDK.
 * This method is utilized to initialize the Software Development Kit (SDK),
 * typically called when instantiating the SDK API.
 *
 * @param application Application object used for initializing the SDK.
 * @deprecated This method is deprecated.
 * Please use the {@code initSdk} method instead.
 */
@Deprecated
public boolean init(Application application)
```

10.2 Initialize the SDK

代码块

```
/**
 * Initializes the SDK.
 * This method is utilized to initialize the Software Development Kit (SDK),
 * typically called when instantiating the SDK API.
 *
 * @param application The Application object used for initializing the SDK.
 * @return return initialize result,
 * it will be turn if it is successed to return,
 * otherwise is falsed
 */
public boolean initSdk(Application application)
```

~~10.3 Initialize Image Library Default Settings (Deprecated)~~

代码块

```
/**
 * Initializes Image Library Default Settings (Deprecated)
 * This method is used to initialize default settings for image processing,
 * including font path and default font path.
 * Text drawing and 1D barcode rendering require prior
 * initialization of the image library.
 *
 * @param fontFamilyPath Custom font path, used to
 *                       set a custom font path for the image library.
 * @param defaultFamilyPath Default font path, used to as a fallback
 *                          when the image library cannot locate
 *                          fonts in the custom font path.
 * @return Returns the initialization result:
 * 0 - Initialization successful
 * 1 - Error: Invalid custom font path
 * 2 - Error: Invalid default font path
 * 3 - Error: Font configuration file not found in the default path
 * 4 - Error: Font file not found in the default path
 * @deprecated This method is deprecated.
 * Please use the {@code initDefaultImageLibrarySettings} method instead.
 */
@Deprecated
public int initImageProcessingDefault(String fontFamilyPath, String
defaultFamilyPath)
```

10.4 Initialize the default settings of the image library

代码块

```
/**
 * Initializes Image Library Default Settings (Deprecated)
 * This method is used to initialize default settings for image processing,
 * including font path and default font path.
 * Text drawing and 1D barcode rendering require prior
 * initialization of the image library.
 *
 * @param fontFamilyPath Custom font path, used to
 *                       set a custom font path for the image library.
 * @param defaultFamilyPath Default font path, used to as a fallback
 *                          when the image library cannot locate
 *                          fonts in the custom font path.
 * @return Returns the initialization result:
 * 0 - Initialization successful
 * 1 - Error: Invalid custom font path
 * 2 - Error: Invalid default font path
 * 3 - Error: Font configuration file not found in the default path
 * 4 - Error: Font file not found in the default path
 */
public int initDefaultImageLibrarySettings(String fontFamilyPath, String defaultFamilyPath)
```

10.5 ~~Initialize the drawing board (deprecated)~~

代码块

```
/**
 * Initializes the Drawing Board
 * This method is used to initialize the drawing board,
 * preparing it for graphical operations.
 *
 * @param width The width of the drawing board, measured in millimeters.
 * @param height The height of the drawing board, measured in millimeters.
 * @param rotate The rotation angle, optional values: 0, 90, 180, 270,
 *               indicating the rotation degree of the drawing board.
 * @param font The font filename (including file extension) used for
 *             specifying the font when drawing text, supporting custom fonts.
 * @deprecated This method is deprecated.
 * Consider using a newer alternative if available.
 */
@Deprecated
public void drawEmptyLabel(float width, float height, int rotate, String font)
```

10.6 Initialize the Drawing Board

代码块

```
/**
 Initializes the Drawing Board
 * This method is used to initialize the drawing board,
 * preparing it for graphical operations.
 *
 * @param width The width of the drawing board, measured in millimeters.
 * @param height The height of the drawing board, measured in millimeters.
 * @param rotate The rotation angle, optional values: 0, 90, 180, 270,
 *                indicating the rotation degree of the drawing board.
 * @param font The font filename (including file extension) used for
 *              specifying the font when drawing text, supporting custom fonts.
 */
public void drawEmptyLabel(float width, float height, int rotate, List<String> font)
```

10.6 Draw Text

10.6.1 Draw Text

代码块

```
/**
 * Draw Text
 * This method is used to draw text on the drawing board,
 * specifying its position, content, font style, and more.
 * Note:
 * 1. Before calling this method, you must first invoke `initSdk()`,
 * `initDefaultImageLibrarySettings()`, and `drawEmptyLabel()`.
 * Refer to the demo for specifics.
 *
 * @param x                   Horizontal coordinate x in millimeters.
 * @param y                   Vertical coordinate y in millimeters.
 * @param width               Width in millimeters.
 * @param height              Height in millimeters.
 * @param value               The text content.
 * @param fontFamily          Font name (without file extension).
 *                            Leave empty to use the default font.
 * @param fontSize            Font size in millimeters.
 * @param rotate              Rotation angle in degrees,
 *                            can be 0, 90, 180, or 270.
 * @param textAlignHorizontal Horizontal alignment:
 *                            0 - left,
 *                            1 - center,
 *                            2 - right.
 * @param textAlignVertical   Vertical alignment:
 *                            0 - top,
 *                            1 - middle,
 *                            2 - bottom.
 * @param lineModel           Line mode:
 *                            1 - fixed width and height, content adjusts,
 *                            2 - fixed width, height adjusts,
 *                            3 - fixed, ellipsis for overflow,
 *                            4 - fixed, content crops,
 *                            6 - fixed, content scales
 *                            if larger than given dimensions.
 * @param letterSpace         Standard letter spacing in millimeters.
 * @param lineSpace           Line spacing (multiplier), in millimeters.
 * @param mFontStyles         Font styles, an array of 4 booleans:
 *                            bold (effective),
 *                            italic (effective),
 *                            underline (effective),
 *                            strikethrough (not effective yet).
 */
public void drawLabelText(float x, float y, float width, float height, String value, String fontFamily, float fontSize, int rotate, int textAlignHorizontal, int textAlignVertical, int lineModel, float letterSpace, float lineSpace, boolean[] mFontStyles)
```

10.6.1 Draw text (supports setting word wrapping mode)

代码块

```


/**
 * Draw text
 * This method is used to draw text on the drawing board, including information such as position, content, font style, etc.
 * Precautions:
 * 1. Before calling this method, you must first call the `initSdk()`, `initDefaultImageLibrarySettings()`, and `drawEmptyLabel()` methods. For details, refer to the DEMO.
 *
 * @param x                   Horizontal coordinate x, unit: millimeters
 * @param y                   Vertical coordinate y, unit: millimeters
 * @param width               Width, unit: millimeters
 * @param height              Height, unit: millimeters
 * @param value               Text content
 * @param fontFamily          Font name (without file name suffix), use the default font when passing an empty string
 * @param fontSize            Font size, unit: millimeters
 * @param rotate              Rotation angle, optional values: 0, 90, 180, 270
 * @param textAlignHorizontal Horizontal alignment: 0 - left alignment, 1 - center alignment, 2 - right alignment
 * @param textAlignVertical   Vertical alignment: 0 - top alignment, 1 - vertical center, 2 - bottom alignment
 * @param lineModel           Line mode: 1 - fixed width and height, content size self-adapts; 2 - fixed width, height self-adapts; 3 - fixed width and height, exceeding content is represented by ellipsis; 4 - fixed width and height, exceeding content is directly cropped; 6 - fixed width and height, content automatically shrinks when exceeding the preset width and height
 * @param letterSpace         Standard spacing between letters, unit: millimeters
 * @param lineSpace           Line spacing (multiple), unit: millimeters
 * @param mFontStyles         Font style, array length is 4, representing in order: bold (effective), italic (effective), underline (effective), strikethrough (temporarily ineffective)
 * @param letterBrokeModel    Word line break mode, this parameter is only valid for English text in the printed content
 * 0, Letter-level line break mode
 * - Line break mode splits text with English letters as the smallest unit
 * - When the remaining space in the line is insufficient, letters are filled one by one to the end of the line, and the remaining letters automatically wrap to the next line
 * - Does not preserve word integrity, may cause words to be truncated (e.g., hello is split into hell and o)
 * 1, Word-level line break mode
 * - Splits text with complete English words as the smallest unit
 * - When the line cannot accommodate a complete word, the whole word moves to the next line
 * - Preserves word integrity, avoids line breaks in the middle of words (e.g., example will not be split into examp and le)
 */
public void drawLabelText(float x, float y, float width, float height, String value, String fontFamily, float fontSize, int rotate, int textAlignHorizontal, int textAlignVertical, int lineModel, float letterSpace, float lineSpace, boolean[] mFontStyles,int letterBrokeModel)
```

10.7 Draw a one-dimensional barcode

代码块

```java
/**
 * Draw a one-dimensional barcode
 * This method is used to draw a one-dimensional barcode on the drawing board, including information such as position, type, and content.
 *
 * @param x Horizontal coordinate x, unit: millimeter
 * @param y Vertical coordinate y, unit: millimeter
 * @param width Width, unit: millimeter
 * @param height Height, unit: millimeter
 * @param codeType One-dimensional barcode type, optional values: 20-CODE128, 21-UPC-A, 22-UPC-E, 23-EAN8, 24-EAN13, 25-CODE93, 26-CODE39, 27-CODEBAR, 28-ITF25
 * @param value One-dimensional barcode content
 * @param fontSize Font size, unit: millimeter
 * @param rotate Rotation angle, optional values: 0, 90, 180, 270
 * @param textHeight Text height, unit: millimeter
 * @param textPosition Text position, optional values: 0-display below, 1-display above, 2-do not display
 */
public void drawLabelBarCode(float x, float y, float width, float height, int codeType, String value, float fontSize, int rotate, float textHeight, int textPosition)
```

10.8 Draw QR Code

10.8.1 Draw QR Code

代码块

```java
/**
 * 绘制二维码
 * 该方法用于在绘图板上绘制二维码，包括位置、类型、内容等信息。
 *
 * @param x        水平坐标x，单位：毫米
 * @param y        垂直坐标y，单位：毫米
 * @param width    宽度，单位：毫米
 * @param height   高度，单位：毫米
 * @param value    二维码内容
 * @param codeType 二维码类型，可选值：31-QR_CODE，32-PDF417，33-DATA_MATRIX，34-AZTEC
 * @param rotate   旋转角度，可选值：0、90、180、270
 */
public void drawLabelQrCode(float x, float y, float width, float height, String value, int codeType, int rotate)
```

10.8.2 Draw QR Code (Supports Logo and Error Correction Level Settings)

代码块

```
/**
 * Draw a QR code
 * This method is used to draw a QR code on the drawing board, including information such as position, type, content, etc.
 *
 * @param x         Horizontal coordinate x, unit: millimeter
 * @param y         Vertical coordinate y, unit: millimeter
 * @param width     Width, unit: millimeter
 * @param height    Height, unit: millimeter
 * @param value     QR code content
 * @param codeType QR code type, optional values: 31-QR_CODE, 32-PDF417, 33-DATA_MATRIX, 34-AZTEC
 * @param rotate    Rotation angle, optional values: 0, 90, 180, 270
 * @param correctLevel   Error correction level, optional values: 1-L, 2-M, 3-Q, 4-H
 * @param qrcodeImageData   Base64 data of the logo, the data header needs to be removed
 * @param anchor    Position of the logo, optional values: 0: centered, 3: bottom right
 * @param scale   Proportion of the QR code occupied
 */
public void drawLabelQrCode(float x, float y, float width, float height, String value, int codeType, int rotate,int correctLevel,String qrcodeImageData, int anchor,float scale)
```

10.9 Drawing Graphics

代码块

```
/**
 * Draw a graph
 * This method is used to draw a graph on the drawing board, including information such as position, type, line style, etc.
 *
 * @param x            Horizontal coordinate x, unit: millimeter
 * @param y            Vertical coordinate y, unit: millimeter
 * @param width        Width, unit: millimeter
 * @param height       Height, unit: millimeter
 * @param graphType    Graph type, optional values: 1 - circle, 2 - ellipse, 3 - rectangle, 4 - rounded rectangle
 * @param rotate       Rotation angle, optional values: 0, 90, 180, 270
 * @param cornerRadius Rounded corner radius, unit: millimeter
 * @param lineWidth    Line width, unit: millimeter
 * @param lineType     Line type, optional values: 1 - solid line, 2 - dashed line type (solid-to-empty ratio 1:1)
 * @param dashWidth    Dashed line style, the array contains the length of the solid segment and the length of the empty segment
 */
public void drawLabelGraph(float x, float y, float width, float height, int graphType, int rotate, float cornerRadius, float lineWidth, int lineType, float[] dashWidth)
```

10.10 Draw Images

代码块

```
/**
 * Draw an image
 * This method is used to draw an image on the drawing board, including information such as the image's Base64 data, position, size, rotation angle, etc.
 *
 * @param imageData            The Base64 data of the image (without the data header)
 * @param x                    Horizontal coordinate x, unit: millimeters
 * @param y                    Vertical coordinate y, unit: millimeters
 * @param width                Width, unit: millimeters
 * @param height               Height, unit: millimeters
 * @param rotate               Rotation angle, optional values: 0, 90, 180, 270
 * @param imageProcessingType  Processing algorithm, default value is 1
 * @param imageProcessingValue Threshold, default value is 127
 */
public void drawLabelImage(String imageData, float x, float y, float width, float height, int rotate, int imageProcessingType, float imageProcessingValue)
```

10.11 Draw Lines

代码块

```java
/**
 * Draw a line
 * This method is used to draw a line on the drawing board, including information such as position, size, rotation angle, line type, etc.
 *
 * @param x Horizontal coordinate x, unit: millimeter
 * @param y Vertical coordinate y, unit: millimeter
 * @param width Width, unit: millimeter
 * @param height Height, unit: millimeter
 * @param rotate Rotation angle, optional values: 0, 90, 180, 270
 * @param lineType Line type, 1: solid line, 2: dashed line type (solid-to-empty ratio 1:1)
 * @param dashWidth Width when the line is a dashed line, including the length of the solid segment and the length of the empty segment, unit: millimeter
 */
public void drawLabelLine(float x, float y, float width, float height, int rotate, int lineType, float[] dashWidth)
```

10.12 Generate and print JSON data

代码块

```java
/**
 * Generate JSON data for tags
 * This method is used to generate JSON data for tags.
 *
 * @return Returns the generated JSON data for tags
 */
public byte[] generateLabelJson()
```