iOS SDK API Documentation V3.2.8 (English Translation)

Title Page

iOS SDK API Documentation V3.2.8

Table of Contents

Revision History

No.	Version	Description of Changes	Author(s)	Date
01	v1.0.1	Document creation	Wang Yadong	2019-03-09
02	v2.0.0	1. Added D11 series printer support 2. Added B21 series printer support 3. Added B3S series printer support 4. Added P1 series printer support	Yu Mingyi / Zhang Bin	2020-02-04
03	v2.0.1	Firmware upgrade optimization, barcode optimization	Yu Mingyi	2020-03-30
04	V2.1.1	B21 Wi-Fi function, P1 printer	Yu Mingyi	2020-04-21
05	v2.1.2	1. P1: Added total sheet count instruction 2. P1: Added blank page printing 3. P1: Added callback for ribbon usage after printing is complete	Yu Mingyi	2020-05-08
06	v2.1.3	1. Adapted for B16 2. Unified error codes 3. P1 panel upgrade and detail optimization	Yu Mingyi	2020-07-07
07	v3.0.0	1. Added B50 series printer support 2. Added B3 series printer support 3. Image phase two access	Yu Mingyi	2020-07-20
08	v3.0.1	1. Added B3S 4.0.1 adaptation 2. Added B16 adaptation 3. Added adaptation for unsupported instructions 4. Second parameter for printing changed	Yu Mingyi	2020-09-01
09	v3.1.0	1. Added adaptation for B32 model 2. Added logging function 3. Added print cache, up to 5 pages of data	Yu Mingyi	2020-09-14
10	v3.1.1	1. Adapted for D110 2. Added adaptation for 8761 Bluetooth chip, needs to be enabled	Yu Mingyi	2021-03-25
11	V3.1.2	1. Removed adaptation for 8761 Bluetooth interface, defaults to compatibility with all Bluetooth access methods 2. Added third-party calling image library drawing function interface	Yu Mingyi	2021-03-25
12	v3.1.3	1. Adapted for S6 model 2. Adapted 300dpi precision to 11.81 3. Synchronized downwards with new features from 3.1.2*beta6* *4. Adapted new protocol, optimized print timeout issue* *5. For new protocol models printing in background, time interval changed to 1ms* *6. Adapted for Z401/Z401R* *7. D101 cutting changed to center cutting* *8. Rolled back image library to 2.0.10*beta4	Yu Mingyi	2021-08-23
13	v3.1.4	1. Adapted private protocol V3, supports sending data while printing 2. Adapted Zhaoxun B3S 3. Adapted for B18	Yu Mingyi	2021-09-26
14	V3.1.5	1. Adapted for B203 model 2. Added B203 data sending interval control instruction 3. Third-party machine adaptation for printing bitmap binarized data	Yu Mingyi	2021-11-10
15	V3.1.6	1. Adapted for B3S and B32 new models	Zhang Bin	2022-07-

				15
16	v3.1.7	1. Adapted P1S new protocol printing 2. Added adaptation for B32/A63 new protocol models, black mark paper cut 1mm from top and bottom 3. Adapted H1S dashed line setting	Yu Mingyi	2022-08-01
17	v3.1.8	1. Adapted dual-color printing protocol	Yu Mingyi	2022-11-01
18	v3.1.9	1. Adapted for K3, K3_W new models 2. Added Wi-Fi related interfaces 3. Removed deprecated interfaces	Yu Mingyi	2023-08-07
19	v3.2.0	1. Removed deprecated interfaces	Yu Mingyi	2023-10-24
20	v3.2.0	1. Added interface to set SDK cache to solve abnormal callback issue after submitting tasks during batch printing	Zhang Bin	2024-01-03
21	v3.2.4	New Features: 1. Support M2 model 2. Support B21*H model* *3. Support B3S*P, A8*P, S6*P models 4. Support B31 model 5. Replaced with new image library b. Improved functionalities of old image library that were not effective c. Fixed some bugs in old image library 6. Support for some models to get the label size installed inside the printer (M2)	Yu Mingyi	2024-07-19
22	v3.2.8	New Features: 1. Support B21 Pro model 2. Support K2 model Bug Fixes: 1. Text wrapping mode 6 inconsistent with other ends	Yu Mingyi	2024-03-15

Product Purpose

The JCAPI interface method documentation is designed to explain the interface methods for standard label drawing output, allowing developers to use these interfaces during secondary development to shorten development time and accelerate development speed.

Product Features

The JCAPI interface provides users with an easy method to create standard label drawing operations. This interface offers methods for creating text, 1D barcodes, 2D barcodes, images, and various shapes, while also supporting rotation of drawings. Users can also use methods to get a preview of the label image created by drawing to assist in making label operations more convenient.

1. Engineering Configuration

1.1 Import Dependencies

Project structure of the DemoToSDK example:

```
DemoToSDK
└── DemoToSDK
    ├── MBProgressHUD
    │   ├── MBProgressHUD.h
    │   ├── MBProgressHUD.m
    │   ├── MBProgressHUD+Extension.h
    │   └── MBProgressHUD+Extension.m
    ├── AppDelegate.h
    ├── AppDelegate.m
    ├── SceneDelegate.h
    ├── SceneDelegate.m
    ├── ViewController.h
    ├── ViewController.m
    ├── TabController.h
    ├── TabController.m
    ├── Model
    │   ├── Model.h
    │   └── Model.m
    ├── JCYMYModels            ←── Image processing
    ├── libSkiaRenderLibrary.a
    ├── JCAPI                  ←── Interface header file
    ├── libJCAPI.a
    ├── libJCLPAPI.a           ←── Printing library
    ├── Cell
    │   ├── Cell.h
    │   └── Cell.m
    ├── Main
    ├── Assets
    ├── LaunchScreen
    ├── Info
    └── main.m
    ├── Products
    └── Frameworks
```


DemoToSDK
DemoToSDK
MBProgressHUD
MBProgressHUD
MBProgressHUD
MBProgressHUD+Extension
MBProgressHUD+Extension
AppDelegate
AppDelegate
SceneDelegate
SceneDelegate
ViewController
ViewController
TabController
TabController
Model
Model
JCYMYModels
libSkiaRenderLibrary.a
JCAPI
libJCAPI.a
libJCLPAPI.a
Cell
Cell
Main
Assets
LaunchScreen
Info
main
Products
Frameworks
图像处理
接口头文件
打印库

1.2 Project Configuration

1.2.1 OC Reflection Configuration

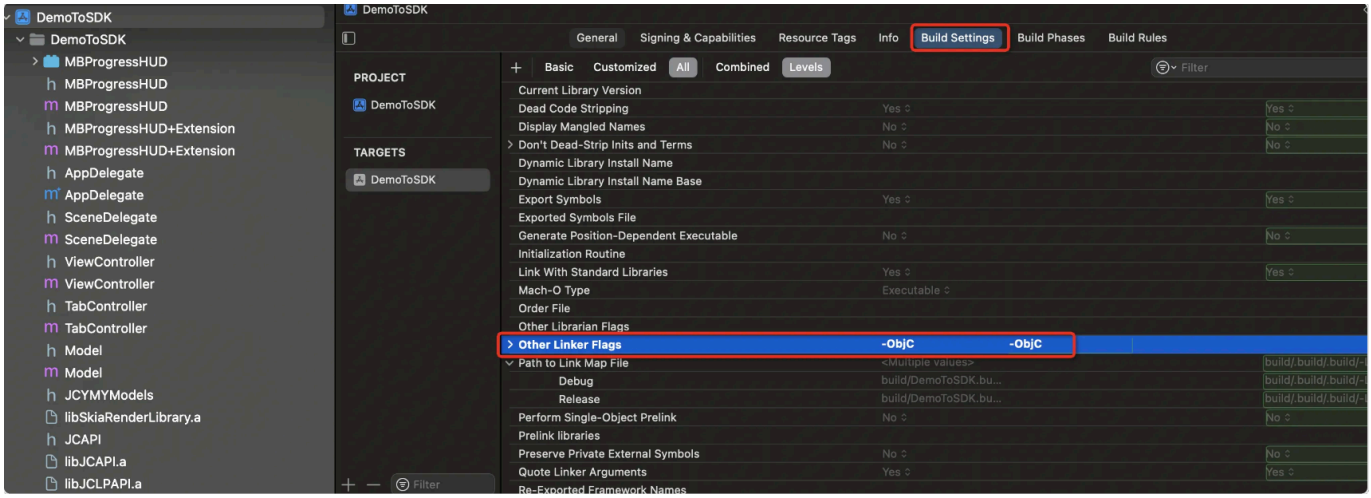
In Xcode, navigate to the project's Build Settings:

```
Project Navigator > DemoToSDK > Build Settings tab > All/Levels view
```

Under the "Linking" section, find "Other Linker Flags" and add:

```
-ObjC
```

This flag is required for the SDK to properly load Objective-C categories and classes through reflection.



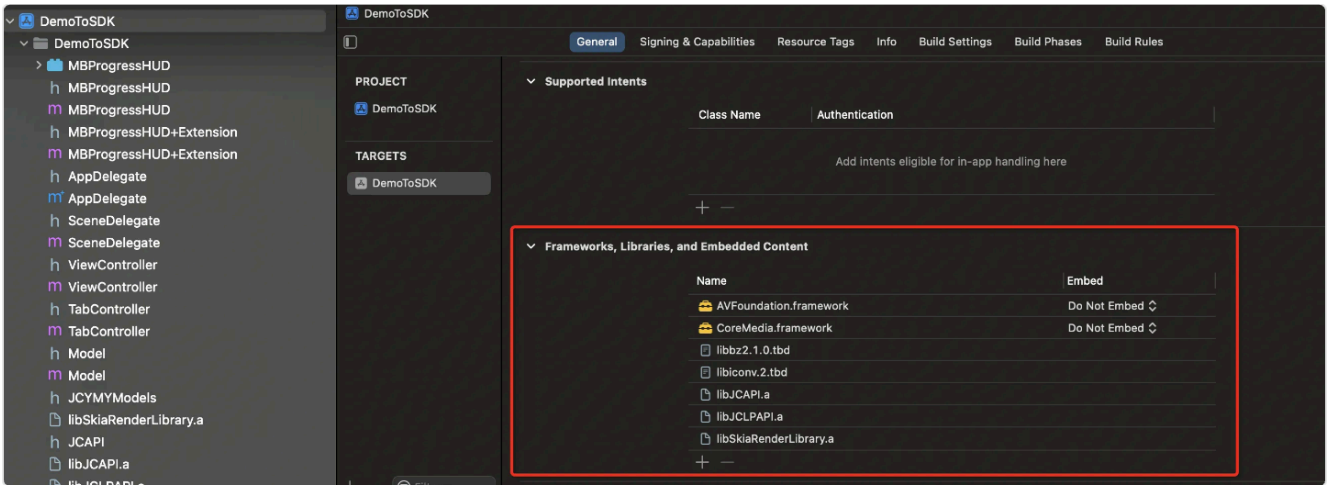
1.2.2 Dependency Configuration

Two main configuration areas need to be set up in Xcode:

1. **Frameworks, Libraries, and Embedded Content**

   Navigate to: `Project Navigator > DemoToSDK > General tab > Frameworks, Libraries, and Embedded Content section`

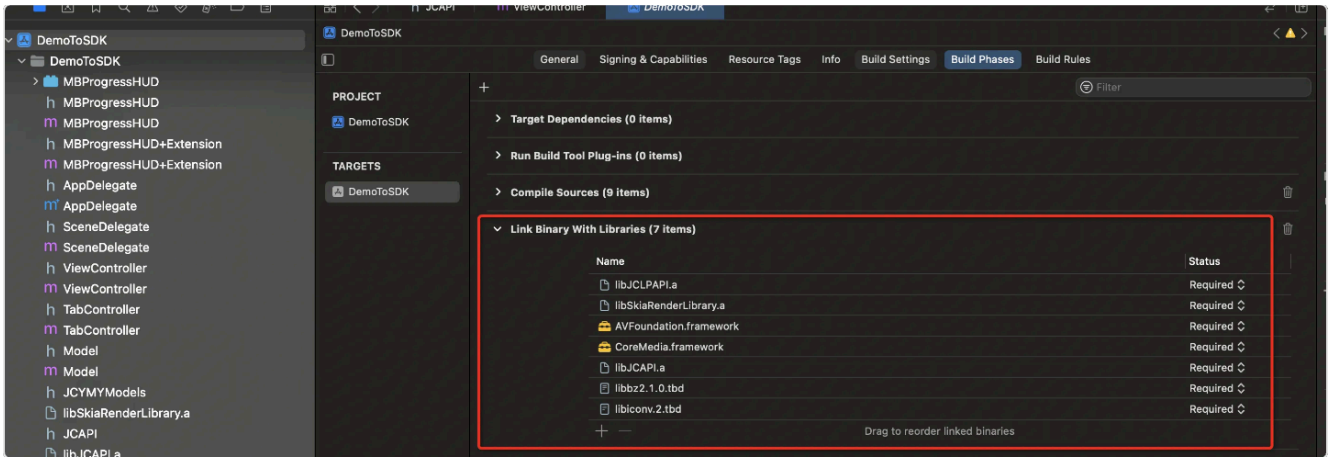
   Add the following frameworks and libraries:
   `- AVFoundation.framework (Do Not Embed) - CoreMedia.framework (Do Not Embed) - libbz2.1.0.tbd - libiconv.2.tbd - libJCAPI.a - l`



2. **Link Binary With Libraries**

   Navigate to: `Project Navigator > DemoToSDK > Build Phases tab > Link Binary With Libraries section`

   Ensure all the following libraries are listed as "Required":
   `- libJCLPAPI.a - libSkiaRenderLibrary.a - AVFoundation.framework - CoreMedia.framework - libJCAPI.a - libbz2.1.0.tbd - libiconv`



These configurations ensure that the SDK libraries and their dependencies are properly linked to your project.

1.2.3 Permission Declaration

You need to declare Bluetooth permissions in the project's info.plist file, for example:

```
Privacy - Bluetooth Always Usage Description   String   This application requires Bluetooth permission when connecting to the pri
NSBluetoothPeripheralUsageDescription         String   This application requires Bluetooth permission when connecting to the print
```

2. Interface Usage Process Example Code

2.1 Search for Printer

2.1.1 Search for Bluetooth Printer

```
// Method for scanning Bluetooth printers
[JCAPI scanBluetoothPrinter:^(NSArray *scanedPrinterNames) {
    // Remove existing data
    [weakSelf.datas removeAllObjects];

    // Loop through scanned Bluetooth printer names
    for (NSString *name in scanedPrinterNames) {
        // Check name length, skip if not in specified range
        if (name.length < 6 || name.length > 20) {
            continue;
        }

        // Create a Model object and set its properties
        Model *m = [[Model alloc] init];
        m.name = name;

        // Add Model object to data array
        [weakSelf.datas addObject:m];
    }
}];
```

2.1.2 Search for WiFi Printer

```
// Method for scanning Wi-Fi printers
[JCAPI scanWifiPrinter:^(NSArray *scanedPrinterNames1) {
    // Hide progress indicator (hud)
    [weakSelf.hud hideAnimated:NO];
    UIStoryboard *main = [UIStoryboard storyboardWithName:@"Main" bundle:nil];
    // Instantiate a "TabController" view controller
    TabController *vc = [main instantiateViewControllerWithIdentifier:@"TabController"];
    // Pass data (weakSelf.datas) to the view controller
    vc.datas = weakSelf.datas;
    // Add this view controller as a child view controller to the current view controller
    [weakSelf addChildViewController:vc];
    // Add the view controller's view to the current view
    [weakSelf.view addSubview:vc.view];
}];
```

2.2 Connect to Printer

2.2.1 Connect to Bluetooth Printer

```
// Connect to a Bluetooth printer with specified name
[JCAPI openPrinter:connectName completion:^(BOOL isSuccess) {
    NSLOG(@"Connection-%@", isSuccess?@"successful":@"failed");
}];
```

2.2.2 Connect to WiFi Printer

```
// Connect to a WiFi printer with specified IP (replace with your printer's actual IP address)
NSString *printerIP = @"192.168.1.100";
[JCAPI openPrinterHost:printerIP completion:^(BOOL isSuccess) {
    NSLOG(@"Connection-%@", isSuccess?@"successful":@"failed");
}];
```

2.3 Printer Usage Related Methods

```
//Font name
NSString *fontName = @"SourceHanSans-Regular";
//Font extension
NSString *fontExtension = @"ttc";
//Font path
NSString *fontPath = [[NSBundle mainBundle] pathForResource:fontName ofType:fontExtension];
//Initialize image library
[JCAPI initImageProcessing:fontPath error:nil];
//Get printing error information
[JCAPI getPrintingErrorInfo:^(NSString *printInfo) {
    if([printInfo isEqualToString:@"19"]){
        //Paper error, can continue printing without handling
    }
    else{
        weakSelf.msgView.hidden = YES;
        MBProgressHUD *hub = [MBProgressHUD showMessage:printInfo];
        [hub hideAnimated:YES afterDelay:2.f];
    }
}];
//Total print count
int count = 1;
//Get printing progress
[JCAPI getPrintingCountInfo:^(NSDictionary *printDicInfo) {
    NSString *totalCount = [printDicInfo valueForKey:@"totalCount"];
    if(totalCount.intValue == count){
        [JCAPI endPrint:^(BOOL isSuccess) {
            if(isSuccess)
                weakSelf.msgView.hidden = YES;
        }];
    }
}];
//Set SDK cache
[JCAPI setPrintWithCache:YES];
//Set total print quantity
[JCAPI setTotalQuantityOfPrints:count];
//Density
int blackRules = 3;
//Paper type
int paperStyle = 1;
//Start print job
[JCAPI startJob:blackRules withPaperStyle:paperStyle withCompletion:^(BOOL isSuccess) {
    if(isSuccess){
        //Set canvas dimensions
    }
}];
```

3. Printer Connection Related Methods

3.1 Search for Bluetooth Printer

```
/**
 * Scan for nearby Bluetooth printers.
 *
 * This method is used to scan for nearby Bluetooth printers and returns a list of discovered printer names via the callback.
 *
 * @param completion Scanning completion callback. After scanning is complete, this callback will be invoked and pass the list of
 *
 * The array 'scanedPrinterNames' contains the names of detected Bluetooth printers. If no printers are detected, the array will
 */
+ (void)scanBluetoothPrinter:(void(^)(NSArray *scanedPrinterNames))completion;
```

3.2 Connect to Bluetooth Printer

```
/**
 * Connect to a Bluetooth printer with the specified name.
 *
 * This method is used to establish a connection with a Bluetooth printer with the specified name. Connection status changes will
 *
 * @param printerName The name of the Bluetooth printer to connect to.
 * @param completion Connection status callback. When the connection status changes, this callback will be used to transmit the co
 *
 * The parameter 'isSuccess' indicates whether the printer connection was successful, YES means connection successful, NO means co
 */
+ (void)openPrinter:(NSString *)printerName
         completion:(DidOpened_Printer_Block)completion;
```

3.3 Search for WiFi Printer

```
/**
 * Scan for nearby Wi-Fi printers.
 *
 * This method is used to scan for nearby Wi-Fi printers and returns a list of discovered printer information via the callback.
 *
 * @param completion Scanning completion callback. After scanning is complete, this callback will be invoked and pass the list of
 *
 * The array 'scanedPrinterNames' contains information about the detected Wi-Fi printers. Each element is a dictionary containing
 *   - 'ipAdd': The IP address of the printer.
 *   - 'bleName': The Bluetooth name.
 *   - 'port': The connection port.
 *   - 'availableClient': Available client connection count.
 */
+ (void)scanWifiPrinter:(void(^)(NSArray *scanedPrinterNames))completion;
```

3.4 Search for WiFi Printer (with Timeout Settings)

```
/**
 * Scan for nearby Wi-Fi printers.
 *
 * This method is used to scan for nearby Wi-Fi printers within a specified timeout period and returns a list of discovered printe
 *
 * @param timeout Scanning timeout in seconds. The scanning operation will be performed within this time period.
 * @param completion Scanning completion callback. After scanning is complete, this callback will be invoked and pass the list of
 *
 * The array 'scanedPrinterNames' contains the names of detected Wi-Fi printers. If no printers are detected, the array will be em
 */
+ (void)scanWifiPrinter:(float)timeout withCompletion:(void(^)(NSArray *scanedPrinterNames))completion;
```

3.5 Get WiFi Name of Currently Connected Phone

```
/**
 * Get the WiFi name currently connected to the phone.
 *
 * This method is used to get the name of the WiFi network that the phone is currently connected to.
 *
 * @return Returns the WiFi name currently connected to the phone.
 */
+ (NSString *)connectingWifiName;
```

3.6 Configure Printer to Connect to Phone's Current WiFi

```
/**
 * Configure printer to connect to the phone's current WiFi.
 *
 * @param wifiName WiFi account name (optional)
 * @param password WiFi password
 * @param completion Callback for whether the printer was successfully configured to connect to WiFi.
 */
+ (void)configurationWifi:(NSString *)wifiName
                 password:(NSString *)password
               completion:(PRINT_DIC_INFO)completion;
```

3.7 Get Printer Wi-Fi Configuration Information

```
/**
 * Get printer Wi-Fi configuration information.
 *
 * This method is used to retrieve Wi-Fi configuration information, usually returning the Wi-Fi name.
 *
 * @param completion Wi-Fi name callback.
 */
+ (void)getWifiConfiguration:(PRINT_DIC_INFO)completion;
```

3.8 Connect to WiFi Printer

```
/**
 * Connect to a printer with the specified IP for Wi-Fi connection.
 *
 * This method is used to establish a Wi-Fi connection with a printer at the specified IP address. Connection status changes will
 *
 * @param host The IP address of the printer, used to specify which printer to connect to.
 * @param completion Connection status callback. When the connection status changes, this callback will be used to transmit the c
 *
 * The parameter 'isSuccess' indicates whether the printer connection was successful, YES means connection successful, NO means c
 */
+ (void)openPrinterHost:(NSString *)host
             completion:(DidOpened_Printer_Block)completion;
```

3.9 Connect to WiFi Printer (Specific Port)

```
/**
 * Connect to a printer with the specified IP for Wi-Fi connection.
 *
 * This method is used to establish a Wi-Fi connection with a printer at the specified IP address. Connection status changes will
 *
 * @param host The IP address of the printer, used to specify which printer to connect to.
 * @param completion Connection status callback. When the connection status changes, this callback will be used to transmit the c
 *
 * The parameter 'isSuccess' indicates whether the printer connection was successful, YES means connection successful, NO means c
 */
+ (void)openPrinterHost:(NSString *)host
             completion:(DidOpened_Printer_Block)completion;
```

3.10 Close Printer

```
/**
 * Close the currently opened printer connection.
 *
 * This method is used to close the currently established printer connection. After executing this operation, it will trigger the
 *
 * Note: Using this method will interrupt the connection with the printer.
 */
+ (void)closePrinter;
```

3.11 Get Currently Connected Printer Name

```
/**
 * Get the name of the currently connected printer (Bluetooth or Wi-Fi).
 *
 * This method is used to get the name of the printer that is currently connected. For Wi-Fi connections, it returns the IP addres
 *
 * @return The name of the currently connected printer. If no printer is connected, it returns nil.
 */
+ (NSString *)connectingPrinterName;
```

3.12 Get Current Connection Status

```
/**
 * Get the current Bluetooth/Wi-Fi connection status.
 *
 * This method is used to get the Bluetooth and Wi-Fi connection status of the current device.
 *
 * @return Return value is an integer indicating the connection status. 0 means no connection, 1 means connected to Bluetooth, 2 r
 */
+ (int)isConnectingState;
```

4. Printer Parameter Settings and Retrieval Methods

4.1 Printer Error Reporting

```
/**
 * Bluetooth/Wi-Fi printer error reception (used after connection is established).
 *
 * @param error Printer errors:
 *   1: Cover is open
 *   2: Out of paper
 *   3: Low battery
 *   4: Battery error
 *   5: Manual stop
 *   6: Data error (Submission of print data failed - B3/Image generation failed/Data transmission error, printer verification fai
 *   7: Temperature too high
 *   8: Paper output error
 *   9: Printer is busy (currently running or feeding paper/currently printing or out of paper/printer is upgrading firmware)
 *   10: Print head not detected
 *   11: Ambient temperature too low
 *   12: Print head not locked
 *   13: Carbon ribbon not detected
 *   14: Incompatible carbon ribbon
 *   15: Used carbon ribbon
 *   16: Unsupported paper type
 *   17: Failed to set paper size
 *   18: Failed to set print mode
 *   19: Failed to set print density (allowed printing, only reports errors)
 *   20: RFID write failure
 *   21: Margin setting error (margin must be greater than 0, top + bottom margins must be less than the picture height, left + ri
 *   22: Communication error (timeout, print command continuously rejected)
 *   23: Printer disconnected
 *   24: Canvas parameter setting error
 *   25: Rotation angle parameter error
 *   26: JSON parameter error (PC)
 *   27: Paper output error (cover close detection)
 *   28: Paper type check
 *   29: RFID tag printing in non-RFID mode
 *   30: Density setting not supported
 *   31: Unsupported printing mode
 *   32: Label material setting failed (material setting timed out or failed, does not interrupt normal printing)
 *   33: Unsupported label material settings (blocks normal printing)
 *   34: Printer error (blocks normal printing)
 *   35: Cutter error (T2 blocks normal printing)
 *   36: Out of paper (T2 release paper)
 *   37: Printer error (T2 cannot recover through commands, need to manually restart the printer)
 *   50: Invalid label
 *   51: Invalid carbon ribbon and label
 */
+ (void)getPrintingErrorInfo:(PRINT_INFO)error;
```

4.2 Get Printer Page Count

```
/**
 * Number of pages printed by Bluetooth/Wi-Fi printer (only valid for printers, some may be lost, app should maintain state with t
 *
 * @param count Number of pages printed (will not be returned after an error occurs)
 * @{
 *   @"totalCount":@"Total printed sheet count" //Required key
 *   @"pageCount":@"Which copy of PageNo is currently being printed" //Optional
 *   @"pageNO":@"Which page is currently being printed" //Optional
 *   @"tid":@"TID code returned from writing RFID" //Optional
 *   @"carbonUsed":@"Carbon ribbon usage in millimeters" //Optional
 * }
 */
+ (void)getPrintingCountInfo:(PRINT_DIC_INFO)count;
```

4.3 Monitor Printer Status Changes

```
/**
 * Monitor printer status changes
 *
 * @param completion
 * @{
 *   @"1": Cover status - 0 open/1 closed
 *   @"2": Battery level change - 1/2/3/4
 *   @"3": Paper installed - 0 no/1 yes
 *   @"5": Carbon ribbon status - 0 no ribbon/1 has ribbon
 *   @"6": WiFi signal strength
 * }
 * @return Whether monitoring printer status changes is supported: YES: supported, NO: not supported
 */
+ (BOOL)getPrintStatusChange:(PRINT_DIC_INFO)completion;
```

4.4 Get Label Size Installed in Printer

```
/**
 * Get the label size installed in the printer (currently only supported for M2 model, firmware version V1.24 or above)
 * Note: Parameters are only valid when statusCode is 0
 * @{@"statusCode":@"0",
 *   @"result":@{@"gapHeightPixel":arrs[0],//Gap height (black mark height) (in pixels)
 *               @"totalHeightPixel":arrs[1],//Paper height (including gap) (in pixels)
 *               @"paperType":arrs[2],//Paper type: 1: gap paper; 2: black mark paper; 3: continuous paper; 4: fixed-size paper; 5
 *               @"gapHeight":arrs[3],//Gap height (black mark height) (in mm)
 *               @"totalHeight":arrs[4],//Paper height (including gap) (in mm)
 *               @"paperWidthPixel":arrs[5],//Paper width (including gap) (in pixels)
 *               @"paperWidth":arrs[6],//Paper width (including gap) (in mm)
 *               @"direction":arrs[7], //Tail direction 1 up 2 down 3 left 4 right (currently not supported)
 *               @"tailLengthPixel":arrs[8],//Tail length (in pixels)
 *               @"tailLength":arrs[9]}} //Tail length (in mm)
 */
+ (void)getPaperInfo:(PRINT_DIC_INFO)completion;
```

5. Barcode Types and Return Information Description

5.1 Barcode Types

Barcode Name	Enum Value
CODEBAR	JCBarcodeFormatCodebar
Code39	JCBarcodeFormatCode39
Code93	JCBarcodeFormatCode93
Code128	JCBarcodeFormatCode128
EAN-8	JCBarcodeFormatEan8
EAN-13	JCBarcodeFormatEan13
ITF(Interleaved Two of Five)	JCBarcodeFormatITF
UPC-A	JCBarcodeFormatUPCA
UPC-E	JCBarcodeFormatUPCE

6. Drawing/Printing Interfaces

6.1 Initialize Image Library

```
/**
 * Initialize the image library.
 *
 * This method is used to set the path of the font file for subsequent image processing operations.
 *
 * @param fontFamilyPath The complete path of the font file.
 * @param error Pointer to an NSError object to receive error information. If an error occurs when setting the font path, it will
 */
+(void) initImageProcessing:(NSString *) fontFamilyPath error:(NSError **)error;
```

6.2 Start Drawing

```
/**
 * Initialize the drawing board.
 *
 * This method is used to initialize a drawing canvas with specified dimensions and positioning. It sets up the canvas for subsequ
 *
 * @param width Canvas width in millimeters
 * @param height Canvas height in millimeters
 * @param horizontalShift Horizontal offset in millimeters (does not take effect)
 * @param verticalShift Vertical offset in millimeters (does not take effect)
 * @param rotate Rotation angle, typically 0
 * @param fonts Font path, can be empty. If no font needs to be specified, nil can be passed in.
 */
+ (void)initDrawingBoard:(float)width
              withHeight:(float)height
       withHorizontalShift:(float)horizontalShift
         withVerticalShift:(float)verticalShift
                    rotate:(int)rotate
                 fontArray:(NSArray<NSString *> *)fonts;
```

6.3 Draw Text

```
/**
 * Draw text on the label.
 *
 * This method is used to draw text on the drawing board with specified position, dimensions, content, font, font size, rotation a
 *
 * @param x Horizontal coordinate (in millimeters)
 * @param y Vertical coordinate (in millimeters)
 * @param w Width (in millimeters)
 * @param h Height (in millimeters)
 * @param text Text content
 * @param fontFamily Font name
 * @param fontSize Font size
 * @param rotate Rotation angle
 * @param textAlignHorizontal Text horizontal alignment: 0 (left), 1 (center), 2 (right)
 * @param textAlignVertical Text vertical alignment: 0 (top), 1 (vertical center), 2 (bottom)
 * @param lineMode Line wrapping mode:
 *                 1-Fixed height, content size adapts automatically
 *                 2-Fixed width, height adapts automatically
 *                 3-Fixed height, content exceeding the area is shown with ellipsis
 *                 4-Fixed height, content exceeding the area is directly truncated
 *                 6-Fixed height, content exceeding the predicted width automatically shrinks
 * @param letterSpacing Character spacing in millimeters
 * @param lineSpacing Line spacing in millimeters
 * @param fontStyles Font style, an array of Boolean values, typically including bold, italic, underline, strikethrough
 *
 * @return Returns a Boolean value indicating whether the text was successfully drawn
 *
 * @note Before drawing text, make sure the image library has been initialized using the method.
 */
+ (BOOL)drawLableText:(float)x
                withY:(float)y
            withWidth:(float)w
           withHeight:(float)h
           withString:(NSString *)text
       withFontFamily:(NSString *)fontFamily
        withFontSize:(float)fontSize
          withRotate:(int)rotate
withTextAlignHorizontal:(int)textAlignHorizontal
withTextAlignVertical:(int)textAlignVertical
       withLineMode:(int)lineMode
   withLetterSpacing:(float)letterSpacing
    withLineSpacing:(float)lineSpacing
     withFontStyle:(NSArray <NSNumber *>*)fontStyles;
```

6.4 Draw 1D Barcode

```
/**
 * Draw a 1D barcode.
 *
 * This method is used to draw a 1D barcode (bar code) on the drawing board, allowing you to specify the barcode's position, dime
 *
 * @param x Horizontal coordinate (in millimeters)
 * @param y Vertical coordinate (in millimeters)
 * @param w Barcode width (in millimeters)
 * @param h Barcode height (in millimeters) (including text height)
 * @param text Barcode content
 * @param fontSize Text font size
 * @param rotate Rotation angle, only supports 0, 90, 180, 270
 * @param codeType 1D barcode type:
 *                 - 20: CODE128
 *                 - 21: UPC-A
 *                 - 22: UPC-E
 *                 - 23: EAN8
 *                 - 24: EAN13
 *                 - 25: CODE93
 *                 - 26: CODE39
 *                 - 27: CODEBAR
 *                 - 28: ITF25
 * @param textHeight Text height (in millimeters)
 * @param textPosition 1D barcode text display position:
 *                     - 0: Display at the bottom
 *                     - 1: Display at the top
 *                     - 2: No display
 *
 * @return Returns a Boolean value indicating whether the barcode was successfully drawn
 *
 * @note Before drawing a 1D barcode, make sure to initialize the image library using the method.
 */
+ (BOOL)drawLableBarCode:(float)x
                   withY:(float)y
               withWidth:(float)w
              withHeight:(float)h
              withString:(NSString *)text
            withFontSize:(float)fontSize
              withRotate:(int)rotate
            withCodeType:(int)codeType
          withTextHeight:(float)textHeight
       withTextPosition:(int)textPosition;
```

6.5 Draw 2D Barcode (QR Code)

```
/**
 * Draw a 2D barcode.
 *
 * This method is used to draw a 2D barcode on the drawing board, allowing you to specify the 2D barcode's position, dimensions, 
 *
 * @param x Horizontal coordinate (in millimeters)
 * @param y Vertical coordinate (in millimeters)
 * @param w 2D barcode width (in millimeters)
 * @param h 2D barcode height (in millimeters)
 * @param text 2D barcode content
 * @param rotate Rotation angle, only supports 0, 90, 180, 270
 * @param codeType 2D barcode type:
 *                 - 31: QR_CODE
 *                 - 32: PDF417
 *                 - 33: DATA_MATRIX
 *                 - 34: AZTEC
 *
 * @return Returns a Boolean value indicating whether the 2D barcode was successfully drawn
 */
+ (BOOL)drawLableQrCode:(float)x
                  withY:(float)y
              withWidth:(float)w
             withHeight:(float)h
             withString:(NSString *)text
              withRotate:(int)rotate
            withCodeType:(int)codeType;
```

6.6 Draw Line

```
/**
 * Draw a line.
 *
 * This method is used to draw a line on the drawing board, allowing you to specify the line's position, dimensions, rotation angl
 *
 * @param x Horizontal coordinate (in millimeters)
 * @param y Vertical coordinate (in millimeters)
 * @param w Line width (in millimeters)
 * @param h Line height (in millimeters)
 * @param rotate Rotation angle, only supports 0, 90, 180, 270
 * @param lineType Line type:
 *                 - 1: Solid line
 *                 - 2: Dashed line type, dash-to-space ratio 1:1
 * @param dashWidth Width of the dash, an array of two numbers indicating the solid line segment length and empty line segment le
 *
 * @return Returns a Boolean value indicating whether the line was successfully drawn
 */
+ (BOOL)DrawLableLine:(float)x
                withY:(float)y
            withWidth:(float)w
           withHeight:(float)h
           withRotate:(int)rotate
         withLineType:(int)lineType
        withDashWidth:(NSArray <NSNumber *>*)dashWidth;
```

6.7 Draw Shape

```
/**
 * Draw a shape.
 *
 * This method is used to draw a shape on the drawing board, allowing you to specify the shape's position, dimensions, line width,
 *
 * @param x Horizontal coordinate (in millimeters)
 * @param y Vertical coordinate (in millimeters)
 * @param w Shape width (in millimeters)
 * @param h Shape height (in millimeters)
 * @param lineWidth Line width (in millimeters)
 * @param cornerRadius Image corner radius (in millimeters)
 * @param rotate Rotation angle, only supports 0, 90, 180, 270
 * @param graphType Shape type, options: 1-Circle, 2-Oval, 3-Rectangle, 4-Rounded Rectangle
 * @param lineType Line type:
 *                 - 1: Solid line
 *                 - 2: Dashed line type, dash-to-space ratio 1:1
 * @param dashWidth Line width, an array of two numbers indicating the solid line segment length and empty line segment length
 *
 * @return Returns a Boolean value indicating whether the shape was successfully drawn
 */
+ (BOOL)DrawLableGraph:(float)x
                 withY:(float)y
             withWidth:(float)w
            withHeight:(float)h
         withLineWidth:(float)lineWidth
      withCornerRadius:(float)cornerRadius
            withRotate:(int)rotate
        withGraphType:(int)graphType
          withLineType:(int)lineType
         withDashWidth:(NSArray <NSNumber *>*)dashWidth;
```

6.8 Draw Image

```
/**
 * Draw an image.
 *
 * This method is used to draw an image on the drawing board, allowing you to specify the image's position, dimensions, image data
 *
 * @param x Horizontal coordinate (in millimeters)
 * @param y Vertical coordinate (in millimeters)
 * @param w Image width (in millimeters)
 * @param h Image height (in millimeters)
 * @param imageData Image Base64 data (without data header)
 * @param rotate Rotation angle, only supports 0, 90, 180, 270
 * @param imageProcessingType Image processing algorithm (default is 1)
 * @param imageProcessingValue Threshold value (default is 127)
 *
 * @return Returns a Boolean value indicating whether the image was successfully drawn
 */
+ (BOOL)DrawLableImage:(float)x
                 withY:(float)y
             withWidth:(float)w
            withHeight:(float)h
        withImageData:(NSString *)imageData
            withRotate:(int)rotate
withImageProcessingType:(int)imageProcessingType
withImageProcessingValue:(float)imageProcessingValue;
```

6.9 Generate JSON String for Label Data

```
/**
 * Generate a JSON string for label data.
 *
 * This method is used to generate a JSON string for label data, which can be submitted to the printer for printing.
 *
 * @return Returns the generated JSON string for label data.
 */
+ (NSString *)GenerateLableJson;
```

6.10 Get Preview of Content Drawn on Canvas

```
/**
 * Get a preview of the label.
 *
 * This method is used to generate a preview image of the label, allowing you to specify the display scale and error code.
 *
 * @param displayScale Display scale
 * @param error Error code returned. If successful, error is nil.
 *
 * @return Returns the generated label preview image
 */
+ (UIImage *)generateImagePreviewImage:(float)displayScale error:(NSError **)error;
```

6.11 Set SDK Print Cache

```
/**
 * Affects caching and pause functionality. Can cache up to 5 tasks, used to improve continuous printing and enhance printing exp
 * Whether to enable SDK caching: YES: enable, NO: disable
 */
+ (void)setPrintWithCache:(BOOL)startCache;
```

6.12 Set Total Number of Prints

```
/**
 * Input total number of prints before printing.
 *
 * @param totalQuantityOfPrints Set the total number of prints, representing the sum of all pages' print quantities. For example,
 */
+ (void)setTotalQuantityOfPrints:(NSInteger)totalQuantityOfPrints;
```

6.13 Start Print Job

```
/**
 * Prepare print job.
 *
 * This method is used to prepare a print job, set the printing density and paper type, and notify the result through the callbac
 *
 * @param blackRules Printing density setting. Specific values depend on the printer model, consider the following guidelines:
 *                   - B series thermal printer (B3S/B21/B203/B1): Supports range 1~5, default value 3
 *                   - K series thermal printer (K3/K3W): Supports range 1~5, default value 3
 *                   - D series thermal printer (D11/D110/D101): Supports range 1~3, default value 2
 *                   - B16 thermal printer: Supports range 1~3, default value 2
 *                   - Thermal transfer printer Z401/B32: Supports range 1~15, default value 8
 *                   - Thermal transfer printer P1/P1S: Supports range 1~5, default value 3
 *                   - Thermal transfer printer B18: Supports range 1~3, default value 2
 *                   - B11/B50/T7/T8 series: 0 (follows printer settings), 1 (lightest), 6 (normal), 15 (darkest)
 * @param paperStyle Paper type setting. Specific values depend on the printer model, consider the following guidelines:
 *                   - B3S/B21/B203/B1/B16/D11/D110/D101/Z401/B32/K3/K3W/P1/P1S:
 *                     1-Gap paper
 *                     2-Black mark paper
 *                     3-Continuous paper
 *                     4-Fixed length paper
 *                     5-Transparent paper
 *                     6-Label tag
 *                   - B11/B50/T7/T8 series:
 *                     0: Continuous paper
 *                     1: Fixed holes (if fixed holes are not supported, automatically switches to gap paper)
 *                     2: Gap paper
 *                     3: Black mark paper
 * @param completion Print completion callback. When the print job is completed, this callback will be used to transmit the printi
 */
+ (void)startJob:(int)blackRules
  withPaperStyle:(int)paperStyle
  withCompletion:(DidPrinted_Block)completion;
```

6.14 Submit Print Job

```
/**
 * Start print label job.
 *
 * This method is used to submit print data, specify the number of copies and handle the callback.
 *
 * @param printData Print data, typically a JSON string for the label
 * @param onePageNumbers Used to specify the number of copies for the current page. For example, if you need to print 3 pages, wit
 * @param completion Print completion callback, used to handle whether the print job was successful.
 */
+ (void)commit:(NSString *)printData
withOnePageNumbers:(int)onePageNumbers
  withComplete:(DidPrinted_Block)completion;
```

6.15 Submit Print Job (Support Write to RFID)

```
/**
 * Start print label job.
 *
 * This method is used to submit print data, specify the number of copies, write RFID data, and handle the callback.
 *
 * @param printData Print data, typically a JSON string for the label
 * @param onePageNumbers Used to specify the number of copies for the current page. For example, if you need to print 3 pages, wit
 * @param epcCode RFID data to write. Can be nil, indicating not to write RFID data. (Only supported on B32R model)
 * @param completion Print completion callback, used to handle whether the print job was successful.
 */
+ (void)commit:(NSString *)printData
withOnePageNumbers:(int)onePageNumbers
       withEpc:(nullable NSString *)epcCode
  withComplete:(DidPrinted_Block)completion;
```

6.16 End Print Job

```
/**
 * Bluetooth/Wi-Fi printing completed (used after printing is complete).
 *
 * @param completion Printing end callback (will not be returned after an error occurs)
 */
+ (void)endPrint:(DidPrinted_Block)completion;
```

6.17 Cancel Printing

```
/**
 * Bluetooth/Wi-Fi cancel printing (used when printing is not completed).
 *
 * @param completion Printing end callback (will not be returned after an error occurs)
 */
+ (void)cancelJob:(DidPrinted_Block)completion;
```

7. Example Application

Here's a comprehensive example of how to use the SDK for a complete printing workflow:

```
// 1. Initialize
// Define font path
NSString *fontPath = [[NSBundle mainBundle] pathForResource:@"SourceHanSans-Regular" ofType:@"ttc"];
// Initialize image library
[JCAPI initImageProcessing:fontPath error:nil];

// 2. Connect to printer
// For Bluetooth printer
[JCAPI openPrinter:@"NIIMBOT-X1" completion:^(BOOL isSuccess) {
    if (isSuccess) {
        NSLog(@"Connected to printer successfully");
        // Proceed with drawing and printing
    } else {
        NSLog(@"Failed to connect to printer");
    }
}];

// Or for WiFi printer
// [JCAPI openPrinterHost:@"192.168.1.100" completion:^(BOOL isSuccess) {
//     // Handle connection result
// }];

// 3. Set up error and status monitoring
[JCAPI getPrintingErrorInfo:^(NSString *printInfo) {
    NSLog(@"Printer error: %@", printInfo);
}];

[JCAPI getPrintStatusChange:^(NSDictionary *statusInfo) {
    NSLog(@"Printer status changed: %@", statusInfo);
}];

// 4. Configure printing
// Set SDK cache (optional)
[JCAPI setPrintWithCache:YES];
// Set total quantity
int totalCopies = 2;
[JCAPI setTotalQuantityOfPrints:totalCopies];

// 5. Start print job
[JCAPI startJob:3 withPaperStyle:1 withCompletion:^(BOOL isSuccess) {
    if (isSuccess) {
        // 6. Create label content
        [JCAPI initDrawingBoard:50 withHeight:30 withHorizontalShift:0 withVerticalShift:0 rotate:0 fontArray:@[]];

        // Draw text
        [JCAPI drawLableText:5 withY:5 withWidth:40 withHeight:10
                  withString:@"Product: ABC-123"
              withFontFamily:@""
                withFontSize:4
                  withRotate:0
     withTextAlignHorizontal:0
       withTextAlignVertical:0
                withLineMode:0
          withLetterSpacing:0
            withLineSpacing:1
             withFontStyle:@[@0, @0, @0, @0]];

        // Draw barcode
        [JCAPI drawLableBarCode:5 withY:15 withWidth:40 withHeight:10
                    withString:@"123456789012"
                  withFontSize:2
                    withRotate:0
```

```
                  withCodeType:JCBarcodeFormatEan13
                withTextHeight:4
               withTextPosition:2];

        // 7. Submit for printing
        NSString *jsonStr = [JCAPI GenerateLableJson];
        [JCAPI commit:jsonStr withOnePageNumbers:1 withComplete:^(BOOL isSuccess) {
            if (isSuccess) {
                NSLog(@"Print data submitted successfully");
            } else {
                NSLog(@"Failed to submit print data");
            }
        }];

        // 8. Monitor print progress
        [JCAPI getPrintingCountInfo:^(NSDictionary *printDicInfo) {
            NSString *totalCount = [printDicInfo valueForKey:@"totalCount"];
            NSLog(@"Printed %@ of %d", totalCount, totalCopies);

            if ([totalCount intValue] == totalCopies) {
                // 9. End print job when all copies are printed
                [JCAPI endPrint:^(BOOL isSuccess) {
                    if (isSuccess) {
                        NSLog(@"Print job ended successfully");
                    }
                }];
            }
        }];
    }
}];

// 10. Cancel job if needed
// [JCAPI cancelJob:^(BOOL isSuccess) {
//     NSLog(@"Print job canceled: %@", isSuccess ? @"YES" : @"NO");
// }];

// Get a preview of the label (optional)
NSError *error = nil;
UIImage *previewImage = [JCAPI generateImagePreviewImage:1.0 error:&error];
if (previewImage) {
    // Use the preview image (e.g., display it in a UIImageView)
}
```

8. Additional Resources

For more information about using the NIIMBOT iOS SDK, refer to the example code provided in the SDK package. The DemoToSDK directory contains a complete example application that demonstrates all the features of the SDK.

8.1 Tips for Common Issues

1. If you encounter connection issues with Bluetooth printers:
   - Ensure the printer is powered on and in discovery mode
   - Check that the required permissions are properly declared in the Info.plist file
   - Verify that the Bluetooth is enabled on the iOS device
2. For printing quality issues:
   - Adjust the printing density using the `blackRules` parameter in the `startJob:withPaperStyle:withCompletion:` method
   - Ensure the correct paper type is selected
3. For drawing/layout issues:
   - Use the `drawLablePreview:withFileName:withComplete:` method to generate a preview of the label before printing
   - Verify all coordinates and dimensions are correctly specified in millimeters

8.2 Known Limitations

- Some features may only be available on specific printer models
- The SDK cache is limited to storing up to 5 pages of data
- Rotation angles are limited to 0°, 90°, 180°, and 270° for most drawing operations