

# **Hoin Printer**

# **IOS File**

**Shenzhen Hoin Electronic Technology Co. Ltd**

**2018.11.12**

This document is only an example of the swift language.

Import aar package

Open Xcode, create a new project, and add three dynamics: BluetoothKit.framework, CocoaAsyncSocket.framework, and HoinPrinterLib.framework to the Project of Embedded Binaries.

## Initialization instance

```
var mPrinter = HoinPrinter()
mPrinter?.switchMode(mode: mode, cb: self)
```

Parameter explanation:

Mode: mode, type: int, initialization mode, 0 is Bluetooth, 1 is WIFI

Cb: Event callback interface, the type is PrinterCallback. The protocol method of the interface needs to be implemented in the current class. The interface contains information such as connection state callback, error callback, and event callback. For details, see PrinterCallback below.

### **PrinterCallback:**

Interface definition:

```
Public protocol PrinterCallback {
    Func onState(state: Int) //state callback, parameter is status code, refer to status code definition
    Func onError(errorCode: Int, info: String) //Error callback, parameter is error code, refer to error code definition
    Func onEvent(event: PrinterEvent) //Event callback, parameter is PrinterEvent event, refer to PrinterEvent definition
```

### **Status code definition:**

```
Public static let BT_STATE_DISCONNECTED = 0; //Bluetooth disconnected
Public static let BT_STATE_LISTEN = 1; //Bluetooth is listening
Public static let BT_STATE_CONNECTING = 2; //Bluetooth connecting
Public static let BT_STATE_CONNECTED = 3; //Bluetooth connected
Public static let WIFI_STATE_CONNECTED = 4; //wifi connected
Public static let WIFI_STATE_DISCONNECTED = 5; //wifi disconnected
```

### **Error code definition:**

```
Public static let BT_NOT_AVALIBLE = 1000; //Bluetooth is not available on this device
Public static let BT_UNABLE_CONNECT_TO_DEVICE = 1001; //Unable to connect to Bluetooth
Public static let BT_CONNECTION_LOST = 1002; //Unable to connect Bluetooth or Bluetooth disconnect
Public static let CONTEXT_ERROR = 1003; //Context error
```

```

Public static let WIFI_SEND_FAILED = 1004; //WIFI failed to send data
Public static let WIFI_CONNECT_ERROR = 1005; //WIFI connection failed
Public static let USB_NOT_FIND_DEVICE = 1006; //USB connection failed
Public static let USB_NO_PERMISSION = 1007; //USB does not have permission
Public static let BT_NO_PERMISSION = 1008; //Bluetooth has no permissions
Public static let DEVICE_NOT_CONNECTED = 1009; //The device is not connected
Public static let IMAGE_NOT_FONUD = 1010; //The device is not connected
Public static let NULL_POINTER_EXCEPTION = 9999; // null pointer exception
Public static let WIFI_EXCEPTION = 2000; //WIFI general abnormal

```

### ***PrinterEvent:***

#### ***Interface definition:***

```

public class PrinterEvent : NSObject {
    public var event: Int;
    public var object: Any
    init(event: Int, object: Any) {
        self.event = event    //Event type
        self.object = object    //Event with information
    }
}

```

### **Event type definition**

```

// Bluetooth search for device events, object is [BKDiscovery] array type, is an array of
searched Bluetooth devices
Public static let EVENT_FIND_BT_DEVICE = 1;

//Bluetooth search completed event
Public static let EVENT_DISCOVERY_BT_FINISHED = 2;

//WIFI receives the data event, the object is the Data type, and is the received data.
Public static let EVENT_WIFI_RECEIVE_DATA = 4;

//Bluetooth available event
Public static let EVENT_BT_AVAILABLE = 6;

// Find the Bluetooth mac address event, object is String type
Public static let EVENT_BT_FIND_MAC = 7;

```

## Scan Bluetooth

```

mPrinter?.startScan()

```

The scanned device is returned by the onEvent though PrinterCallback

## Stop scanning Bluetooth

```
mPrinter?.stopScan()
```

## Connect device

```
mPrinter?.connect(object);
```

When the mode is Bluetooth, the parameter is BKDiscovery;  
When the mode is WIFI, the parameter is the IP address of the device.

## Disconnect device

```
mPrinter?.disconnect();
```

## Change the printer type (58mm or 80mm)

```
mHoinPrinter.switchType(type58);
```

The parameter is boolean type, true is to switch to 58mm printer, false is to switch to 80mm printer

## Set centering print effect

```
mPrinter?.setCenter(center: mCenter)
```

The parameter is bool type, true is centered, false is not centered

## Print string

```
mPrinter?.printText(text: str, doubleH: mDoubleH, doubleW: mDoubleW, bold: mStrong, center: mCenter)
```

There are five parameters

Parameter 1: Printed text string, the type is String

Parameter 2: Whether to print double high, the type is bool

Parameter 3: Whether to print in double width, the type is bool

Parameter 4: Whether the font is bold, the type is bool

Parameter 5: Whether it is centered, the type is bool

## Print Image

```
mPrinter?.printImage(image: selectedImage)
```

There is one parameter

Parameter 1: Image, type is UIImage

## Print QR code

```
mPrinter?.printQRCode(qrcode: str)
```

There is one parameter

Parameter 1: The printed QR code is the corresponding string content

## Print 1D bar code

```
mPrinter?.printBarCode()
```

## Open cash drawer box

```
mPrinter?.openBox()
```

## Cutter test

```
mPrinter?.testCutting()
```