HSPOS iOS PrinterSDK 使用说明书

项	说明
产品	HSPOS iOS PrinterSDK
平台	iOS 15+
入口类	`PrinterSDK`（`PrinterSDK.shared`）
常量	`PrinterConstants`
连接抽象	`PrinterConnection`
配套文档	《PrinterSDK iOS 函数说明》
英文版	`PrinterSDK_iOS_User_Manual.pdf`

---

1. 产品简介

PrinterSDK 是面向 iOS 的热敏打印机开发库，通过统一入口完成设备发现、连接、打印与状态查询。

主要能力：

- 连接：蓝牙 BLE、网口 TCP（常见端口 9100）
- 标准 ESC/POS 票据打印（文本、条码、二维码、光栅图、切纸等）
- 页模式坐标绘制
- JPL 标签打印
- 状态查询与缓冲清理

业务层建议只使用：`PrinterSDK`、`PrinterConstants`、`PrinterConnection`。

函数与常量的完整说明见《函数说明》文档。

---

2. 系统要求与集成（推荐：三步完成）

2.1 环境

项	要求
最低 iOS	15.0
语言	Swift 5.9+
Xcode	15+（建议）
交付物	`HSPOS-PrinterSDK-iOS-*.zip`（解压后含 `Package.swift`）

2.2 三步集成（客户推荐方式）

第 1 步：把 SDK 加进工程

1. 解压官方交付包 `HSPOS-PrinterSDK-iOS-x.x.x.zip`
2. 打开你的 App 工程（Xcode）
3. 菜单 **File → Add Package Dependencies...**
4. 点 **Add Local...**，选择**解压后的文件夹**（目录内能看到 `Package.swift`）
5. 产品勾选 **PrinterSDK** → **Add Package**

完成后，在需要打印的 Swift 文件顶部：

```
import PrinterSDK
```

第 2 步：配置权限

将交付包内 `Info-Plist-Snippet.plist` 的内容复制到宿主 App 的 `Info.plist`（也可用 Xcode 的 Info 面板添加）：

Key	用途
`NSBluetoothAlwaysUsageDescription`	BLE 扫描 / 连接
`NSBluetoothPeripheralUsageDescription`	兼容旧系统声明
`NSLocalNetworkUsageDescription`	TCP 局域网打印机

示例文案：

- 蓝牙：`需要使用蓝牙连接热敏打印机`
- 本地网络：`需要通过局域网连接网络打印机`

权限由系统弹窗请求；用户也可在「设置」中管理。

第 3 步：调用打印

```
import PrinterSDK

let sdk = PrinterSDK.shared

sdk.connectNetworkPrinterAsync(ip: "192.168.1.100", port: 9100) { conn in
    DispatchQueue.global(qos: .userInitiated).async {
        _ = sdk.initPrinter(conn)
        _ = sdk.setAlignment(conn, align: PrinterConstants.ALIGNMENT_CENTER)
        _ = sdk.printText(conn, text: "Hello PrinterSDK\n")
        _ = sdk.feedAndCutPaper(conn)
        sdk.disconnectPrinter(conn)
    }
} onFailed: { message in
    print(message)
}
```

交付包内还有：

路径	内容
`README.md` / `集成说明.txt`	三步摘要
`Examples/MinimalUsage.swift`	最小示例
`Docs/*.pdf`	说明书与函数说明（中英）
`Sources/PrinterSDK/`	SDK 源码（由 SPM 自动编译）

业务层只使用：`PrinterSDK`、`PrinterConstants`、`PrinterConnection`。

2.3 交付包目录结构（了解即可）

```
HSPOS-PrinterSDK-iOS-x.x.x/
├── Package.swift              ← Add Local 时选中本目录
├── Sources/PrinterSDK/        ← SDK 源码
├── Docs/                      ← PDF 文档
├── Examples/                  ← 示例代码
├── Info-Plist-Snippet.plist   ← 权限片段
├── README.md
└── VERSION.txt
```

请勿在业务层直接依赖内部 Manager 类型，以便后续升级兼容。

---

3. 架构

```
Your iOS App
      │
      ▼  PrinterSDK.shared
┌──────────────────────────────────────┐
│ PrinterSDK                           │
│ 连接 │ 标准 POS │ 页模式 │ 标签 │ 状态 │
└──────┬──────────┬────────┬─────┬─────┘
       ▼          ▼        ▼     ▼
 Communication  PosPrinter Page  Label
 Manager        Manager    Mode  Mode
       │
       ▼
 PrinterConnection (BLE / Network)
       │
       ▼
 物理打印机
```

每次打印前确认：`connection != nil && connection.isConnected()`。

---

4. 快速开始

4.1 初始化

```
let sdk = PrinterSDK.shared
// 无需额外 Context；不再使用时可调用 sdk.release()
```

4.2 最小打印流程（网口）

```
let sdk = PrinterSDK.shared

sdk.connectNetworkPrinterAsync(ip: "192.168.1.100", port: 9100) { conn in
    DispatchQueue.global(qos: .userInitiated).async {
        _ = sdk.initPrinter(conn)
        _ = sdk.setAlignment(conn, align: PrinterConstants.ALIGNMENT_CENTER)
        _ = sdk.printText(conn, text: "Hello PrinterSDK\n")
        _ = sdk.cutPaper(conn, cutType: PrinterConstants.CUT_PARTIAL)
        sdk.disconnectPrinter(conn)
    }
} onFailed: { message in
    print(message)
}
```

4.3 最小打印流程（BLE）

```
final class MyPrinterHost: BluetoothDeviceListener {
    let sdk = PrinterSDK.shared

    func start() {
        sdk.addBluetoothListener(self)
        _ = sdk.startBluetoothDiscovery()
    }

    func onDeviceFound(_ peripheral: CBPeripheral, rssi: Int) {
        if let conn = sdk.connectBluetoothPrinter(peripheral) {
            _ = sdk.initPrinter(conn)
            _ = sdk.printText(conn, text: "BLE OK\n")
        }
    }

    func onDiscoveryFinished() {}
}
```

系统会弹出蓝牙授权对话框。请确认打印机已开机，并支持 HSPOS BLE（GATT Service `FF00`，写特征 `FF02`）。

---

5. 权限配置（Info.plist）

见 **§2.2 第 2 步**。交付包已提供 `Info-Plist-Snippet.plist`，复制到宿主 App 即可。

如使用 Bonjour 发现，可额外声明 `NSBonjourServices`（例如 `_printer._tcp`）。

---

6. 打印机连接

6.1 支持的连接方式

通道	说明
蓝牙 BLE	GATT Service `FF00`，写特征 `FF02`
网口 TCP	常见 RAW 打印端口 **9100**

6.2 BLE

```
sdk.addBluetoothListener(listener)
_ = sdk.startBluetoothDiscovery()
let devices = sdk.getBluetoothDevices()   // [CBPeripheral]
let conn = sdk.connectBluetoothPrinter(peripheral)
sdk.stopBluetoothDiscovery()
sdk.removeBluetoothListener(listener)
```

说明：

- 设备类型为 `CBPeripheral`
- 大批量数据会分片写入；连接层可能合并小写入
- 打印复杂内容后，连接层会按需 flush

6.3 网口

```
// 推荐：异步（可在主线程调用）
sdk.connectNetworkPrinterAsync(ip: "192.168.1.100", port: 9100,
    onSuccess: { conn in /* 回调线程可能不是主线程 */ },
    onFailed: { err in })

// 同步：勿在主线程调用
let conn = sdk.connectNetworkPrinter(ip: "192.168.1.100", port: 9100)
```

6.4 断开与清理

```
sdk.disconnectPrinter(connection)
_ = sdk.clearPrinterBuffer(connection)
_ = sdk.cancelPrint(connection)
sdk.release()
```

---

7. 标准 POS 打印

常用示例：

```
_ = sdk.initPrinter(conn)
_ = sdk.setPrintAreaWidth(conn, width: PrinterConstants.PAPER_WIDTH_80MM)
_ = sdk.setAlignment(conn, align: PrinterConstants.ALIGNMENT_LEFT)
_ = sdk.printTextInGBK(conn, text: "欢迎光临\n")
_ = sdk.printTextLineLeftRight(conn, left: "商品A", right: "$12.00")
_ = sdk.printQRCode(conn, content: "https://hsprinter.com", size: 3)
_ = sdk.feedAndCutPaper(conn)
```

7.1 放大字号（勿混用顺序）

方法	指令	用途
`setChineseDoubleSize`	`FS W n`	中文倍宽倍高开关
`setTextScale`	`GS ! n`	宽高倍率 0–7
`setPrintMode`	`ESC ! n`	打印模式倍宽 / 倍高

打印机以最后收到的相关指令为准。

7.2 多编码文本

- `printText`（默认）
- `printTextInUTF8` / `printTextInGBK` / `printTextInBIG5`
- `printTextInShiftJIS` / `printTextInEUCKR`

中文票据优先使用 `printTextInGBK`。

7.3 光栅图

- `printRasterImageFromFile` / `FromData` / `FromPixels`
- 压缩相关常量含 RLE / JBIG / AUTO；**当前实现统一以 GS v 0 输出**（其它模式回退）

完整参数见《函数说明》。

---

8. 页模式打印

推荐流程：

1. `selectPageMode` / `selectPageModeEx(...)`
2. `setPrintAreaInPageMode`，绘制文本 / 线 / 矩形 / 条码 / 二维码
3. `pagePrintPage`
4. `exitPageMode`

部分机型在页模式内对光栅图支持不稳定；复杂图文可参考 Demo 的 `PageModeExtendedPrinter`（页内文本 + 退出后补打光栅）。

---

9. 标签模式（JPL）

```
guard sdk.enableLabelMode(conn) else { return }
defer { _ = sdk.disableLabelMode(conn) }

_ = sdk.labelPageBegin(conn, x: 0, y: 0, width: 400, height: 300, rotation: 0)
_ = sdk.printLabelTextGBK(conn, text: "标签", x: 20, y: 20, fontHeight: 24, fontType: 0)
_ = sdk.printLabelQRCode(conn, data: "ABC123", x: 20, y: 80, size: 4)
_ = sdk.labelPagePrint(conn)
```

注意：

- 标签指令面向**标签打印机**；在普通票据机上可能无输出或返回失败
- 参数非法时返回 `false`，不会导致 App 崩溃
- 标签位图默认按原图宽度缩放（上限 576 点），可用 `targetWidth` 指定宽度

---

10. 状态查询

```
let status = sdk.getPrinterStatus(conn)
let online = sdk.isPrinterOnline(conn)
let paper  = sdk.hasPaper(conn)
let cover  = sdk.isCoverOpen(conn)

_ = sdk.queryRTStatus(conn, type: PrinterConstants.STATUS_TYPE_PRINTER)
_ = sdk.queryTransmitStatus(conn, type: PrinterConstants.STATUS_TYPE_OFFLINE)
```

状态位：`STATUS_ONLINE` / `STATUS_PAPER_OUT` / `STATUS_COVER_OPEN`。

BLE 通知回包可能有 1–2 秒延迟，超时请勿设得过短。

---

11. 常用常量

常量	值	说明
`ALIGNMENT_LEFT/CENTER/RIGHT`	0/1/2	对齐
`CUT_FULL` / `CUT_PARTIAL`	0/1	全切 / 半切
`PAPER_WIDTH_58MM`	384	58mm 可打宽度（点）
`PAPER_WIDTH_80MM` / `PAPER_WIDTH_72MM`	576	80mm 有效宽约 72mm
`CHAR_COLUMNS_72MM`	48	Font A 下约 48 列

更多常量见《函数说明》第 3 章。

---

12. Demo 与示例

仓库内 Demo（`ContentView` + `Demo/Print/*`）提供：

- 连接：BLE / 网口
- 测试：58/80mm 票据、图形票据、光栅、表格、页模式、标签、状态等

打印任务建议放在后台队列执行（见 `PrintJobRunner`），避免阻塞 UI。

---

13. FAQ

Q: 连接不上蓝牙?

A: 检查系统蓝牙开关与 App 权限；确认机型支持 BLE GATT `FF00/FF02`；靠近后重新扫描。

Q: 网口连接超时?

A: 确认手机/平板与打印机同一局域网，IP 与端口（多为 9100）正确，并允许本地网络权限。

Q: 标签相关按钮在票据机上没反应?

A: 标签 JPL 面向标签机。请确认机型与纸张类型；Demo 会对失败给出提示。

Q: 中文乱码?

A: 优先 `printTextInGBK`，或先 `setDoubleByteEncoding(..., GBK)`。页模式多编码依赖机型固件支持。

Q: BLE 打印慢或卡住?

A: 确认写特征可用且权限已授权；大批量数据由 SDK 分片。设备若不支持 notify，状态查询可能失败，但不影响单纯下发打印。

---

14. 文档与交付物

内容	用途
本文《使用说明书》	集成、连接、典型打印流程
《函数说明》	API 签名、参数、返回值与常量
`HSPOS-PrinterSDK-iOS-*.zip`	客户交付包（三步集成）

修订源文：`DOC/PrinterSDK_iOS_使用说明书.md`。
