

TSPL SDK 接口文档

文档修改记录

序号	版本号	修改内容	修改者	评审者	修改日期
1	v1.0.0	初始版本	邱贤	林德春	2026.04.11
2	v1.0.1	新增错误码 UXErrorCode 说明, PrinterError 增加 errorCode 字段	邱贤	林德春	2026.08.04

1. 使用说明

1.1 工程配置说明

在项目的 `build.gradle` 中添加依赖:

```
implementation files('libs/UxPrint-sdk-v1.xx.xx.aar')
implementation files('libs/lzo-sdk-v1.xx.xx.aar')
```

1.2 权限配置

```
<!-- 基础权限 -->
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<uses-permission android:name="android.permission.BLUETOOTH_CONNECT" />
<uses-permission android:name="android.permission.BLUETOOTH_SCAN" />
```

1.3 混淆配置

在 `proguard-rules.pro` 中添加:

```
-keep class com.printer.sdk.** { *; }
-keep interface com.printer.sdk.** { *; }
```

1.4 初始化

```
// 创建连接
val connection = BluetoothConnection()
connection.connect(ConnectionParams.Bluetooth(address = "00:11:22:33:44:55"))

// 创建 TsplPrinter 实例
val printer = TsplPrinter(connection)

// 设置标签尺寸（标签打印的前提）
printer.setSize(width = 60, height = 40)
printer.setGap(gap = 2, offset = 0)

// 设置编码（可选）
printer.setEncoding("GBK")
```

2. 接口目录

TSPL SDK 接口文档

文档修改记录

1. 使用说明

1.1 工程配置说明

1.2 权限配置

1.3 混淆配置

1.4 初始化

2. 接口目录

3. setSize

4. setGap

5. setBlackMark

6. setOffset

7. setReference

8. setDirection

9. setSpeed

10. setDensity

11. setCodePage

12. setEncoding

13. setBold

14. setBlackMarkPosition

15. printText

16. printBarcode

17. printQRCode

18. printImage

19. drawBox

20. drawBar

21. drawCircle

22. drawEllipse

- 23. erase
- 24. reverse
- 25. printPDF417
- 26. printDataMatrix
- 27. printMaxiCode
- 28. putPcx
- 29. printWaterMark
- 30. print
- 31. clear
- 32. feedPaper
- 33. backFeed
- 34. limitFeed
- 35. formFeed
- 36. home
- 37. cutPaper
- 38. sound
- 39. selfTest
- 40. 附录
 - 40.1 字体枚举 (TsplFont)
 - 40.2 旋转角度枚举 (Rotation)
 - 40.3 条码类型枚举 (BarcodeType)
 - 40.4 抖动算法枚举 (DitheringAlgorithm)
 - 40.5 错误类型说明

3. setSize

描述：设置标签尺寸。标签打印的前提条件，必须在打印前调用。

方法签名：

```
suspend fun setSize(width: Int, height: Int): Result<Unit>
```

参数：

参数名	类型	必填	说明
width	Int	是	标签宽度（mm）
height	Int	是	标签高度（mm）

示例代码：

```
printer.setSize(width = 60, height = 40)
```

4. setGap

描述：设置标签间隙。

方法签名：

```
suspend fun setGap(distance: Int, offset: Int = 0): Result<Unit>
```

参数：

参数名	类型	必填	默认值	说明
distance	Int	是	-	间隙高度（mm）
offset	Int	否	0	偏移量（mm）

示例代码：

```
printer.setGap(distance = 2, offset = 0)
```

5. setBlackMark

描述：设置黑标模式。

方法签名：

```
suspend fun setBlackMark(height: Int, offset: Int = 0): Result<Unit>
```

参数：

参数名	类型	必填	默认值	说明
height	Int	是	-	黑标高度（mm）
offset	Int	否	0	偏移量（mm）

示例代码：

```
printer.setBlackMark(height = 10, offset = 0)
```

6. setOffset

描述：设置打印偏移。

方法签名：

```
suspend fun setOffset(distance: Int): Result<Unit>
```

参数：

参数名	类型	必填	说明
distance	Int	是	偏移量（mm）

示例代码：

```
printer.setOffset(0)
```

7. setReference

描述：设置参考点。

方法签名：

```
suspend fun setReference(x: Int, y: Int): Result<Unit>
```

参数：

参数名	类型	必填	说明
x	Int	是	X坐标（点）
y	Int	是	Y坐标（点）

示例代码：

```
printer.setReference(x = 0, y = 0)
```

8. setDirection

描述：设置打印方向。

方法签名：

```
suspend fun setDirection(direction: Int): Result<Unit>
```

参数：

参数名	类型	必填	说明
direction	Int	是	打印方向（0=从左到右，1=从右到左），方向值使用 Direction 枚举（见附录 40.2）

示例代码：

```
printer.setDirection(0)
```

9. setSpeed

描述：设置打印速度。

方法签名：

```
suspend fun setSpeed(speed: Int): Result<Unit>
```

参数：

参数名	类型	必填	说明
speed	Int	是	打印速度（1-14，英寸/秒）

示例代码：

```
printer.setSpeed(4)
```

10. setDensity

描述：设置打印浓度。

方法签名：

```
suspend fun setDensity(density: Int): Result<Unit>
```

参数：

参数名	类型	必填	说明
density	Int	是	浓度值（0-15，0最淡，15最深）

示例代码：

```
printer.setDensity(10)
```

11. setCodePage

描述：设置代码页。

方法签名：

```
suspend fun setCodePage(codePage: String): Result<Unit>
```

参数：

参数名	类型	必填	说明
codePage	String	是	代码页名称

支持的代码页：

代码页	说明
PC437	美国标准
PC850	多语言
PC852	拉丁语2
PC860	葡萄牙语
PC863	加拿大法语
PC865	北欧
PC857	土耳其语
PC737	希腊语
ISO-8859-1	西欧（Latin-1）
ISO-8859-2	中欧（Latin-2）
ISO-8859-5	西里尔语
ISO-8859-7	希腊语
ISO-8859-9	土耳其语
GBK	中文简体（GB2312扩展）
GB2312	中文简体
Big5	中文繁体
UTF-8	Unicode

示例代码：

```
printer.setCodePage("GBK")
```

12. setEncoding

描述：设置编码。用于设置打印机与计算机通信时的字符编码。

方法签名：

```
fun setEncoding(encoding: String)
```

参数：

参数名	类型	必填	说明
encoding	String	是	编码格式（如GBK、UTF-8、GB2312等）

示例代码：

```
printer.setEncoding("GBK")
```

13. setBold

描述：设置粗体。

方法签名：

```
suspend fun setBold(bold: Int): Result<Unit>
```

参数：

参数名	类型	必填	说明
bold	Int	是	粗体程度（0-5，0表示关闭，1-5表示加粗程度）

示例代码：

```
printer.setBold(1)
```

14. setBlackMarkPosition

描述：设置黑标位置。

方法签名：

```
suspend fun setBlackMarkPosition(mode: Int): Result<Unit>
```

参数：

参数名	类型	必填	说明
mode	Int	是	模式（0=禁用，1=启用）

示例代码：

```
printer.setBlackMarkPosition(0)
```

15. printText

描述：打印文本。

方法签名：

```
suspend fun printText(  
    text: String,  
    x: Int = 0,  
    y: Int = 0,  
    options: TextOptions = TextOptions()  
): Result<Unit>
```

参数：

参数名	类型	必填	默认值	说明
text	String	是	-	要打印的文本
x	Int	否	0	X坐标（点）
y	Int	否	0	Y坐标（点）
options	TextOptions	否	TextOptions()	文本选项

TextOptions 属性：

属性名	类型	默认值	说明
font	TsplFont	FONT_0	字体
fontSize	Int?	null	字体大小（点）
xMultiplier	Int	1	宽度倍数（1-10）
yMultiplier	Int	1	高度倍数（1-10）
bold	Boolean	false	TSPL不支持
underline	Boolean	false	TSPL不支持
align	TextAlign	LEFT	TSPL不支持
fontSmall	Boolean	false	TSPL不支持
reverse	Boolean	false	TSPL不支持
rotation	Rotation	NORMAL	旋转角度

示例代码：

```
printer.printText(
    text = "TSPL Label",
    x = 50,
    y = 30,
    options = TextOptions(
        font = TsplFont.FONT_0,
        fontSize = 24,
        xMultiplier = 2,
        yMultiplier = 2
    )
)
```

16. printBarcode

描述：打印条码。

方法签名：

```
suspend fun printBarcode(
    data: String,
    type: BarcodeType,
    x: Int = 0,
    y: Int = 0,
    options: BarcodeOptions = BarcodeOptions()
): Result<Unit>
```

参数：

参数名	类型	必填	默认值	说明
data	String	是	-	条码数据
type	BarcodeType	是	-	条码类型
x	Int	否	0	X坐标（点）
y	Int	否	0	Y坐标（点）
options	BarcodeOptions	否	BarcodeOptions()	条码选项

BarcodeOptions 属性：

属性名	类型	默认值	说明
showText	Boolean	true	是否显示条码下方文字
textAbove	Boolean	false	文字是否在条码上方
rotation	Rotation	NORMAL	TSPL不支持

条码类型 (BarcodeType) 详细说明：

值	说明	支持的数据格式
CODE128	Code 128	ASCII字符集
CODE128A	Code 128 A	大写字母和数字
CODE128B	Code 128 B	大小写字母和数字
CODE128C	Code 128 C	纯数字（偶数位）
CODE39	Code 39	字母、数字、特殊字符
CODE93	Code 93	字母、数字、特殊字符
EAN13	EAN-13	12-13位数字
EAN8	EAN-8	7-8位数字
UPC_A	UPC-A	11-12位数字
UPC_E	UPC-E	6-8位数字
CODABAR	Codabar	数字和特殊字符
ITF	ITF	偶数位数字

示例代码：

```
printer.printBarcode(  
    data = "1234567890",  
    type = BarcodeType.CODE128,  
    x = 50,  
    y = 80,  
    options = BarcodeOptions(showText = true)  
)
```

17. printQRCode

描述：打印二维码。

方法签名：

```
suspend fun printQRCode(  
    data: String,  
    x: Int = 0,  
    y: Int = 0,  
    options: QRCodeOptions = QRCodeOptions()  
) : Result<Unit>
```

参数：

参数名	类型	必填	默认值	说明
data	String	是	-	二维码数据
x	Int	否	0	X坐标（点）
y	Int	否	0	Y坐标（点）
options	QRCodeOptions	否	QRCodeOptions()	二维码选项

QRCodeOptions 属性：

属性名	类型	默认值	说明
moduleSize	Int	6	模块大小（1-16，点）
errorCorrection	ErrorCorrectionLevel	M	纠错等级
rotation	Rotation	NORMAL	TSPL不支持

示例代码：

```
printer.printQRCode(  
    data = "https://www.example.com",  
    x = 200,  
    y = 150,  
    options = QRCodeOptions(  
        moduleSize = 8,  
        errorCorrection = ErrorCorrectionLevel.H  
    )  
)
```

18. printImage

描述：打印图片。

方法签名：

```
// 重载1：使用字节数组  
suspend fun printImage(  
    imageData: ByteArray,  
    width: Int,  
    height: Int,  
    x: Int = 0,  
    y: Int = 0,  
    options: ImageOptions = ImageOptions()  
): Result<Unit>
```

```
// 重载2: 使用Bitmap
suspend fun printImage(
    bitmap: Bitmap,
    x: Int = 0,
    y: Int = 0,
    options: ImageOptions = ImageOptions()
): Result<Unit>
```

参数:

参数名	类型	必填	默认值	说明
imageData	ByteArray	是 (重载1)	-	图片字节数据
width	Int	是 (重载1)	-	图片宽度 (点)
height	Int	是 (重载1)	-	图片高度 (点)
bitmap	Bitmap	是 (重载2)	-	要打印的图片
x	Int	否	0	X坐标 (点)
y	Int	否	0	Y坐标 (点)
options	ImageOptions	否	ImageOptions()	图片选项

ImageOptions 属性:

属性名	类型	默认值	说明
threshold	Int	-1	二值化阈值 (-1为自动计算)
algorithm	DitheringAlgorithm	DIFFUSION	抖动算法
maxWidth	Int?	null	最大宽度 (超过将自动缩放, 点)

示例代码:

```
val bitmap = BitmapFactory.decodeResource(resources, R.drawable.logo)
printer.printImage(
    bitmap = bitmap,
    x = 50,
    y = 200,
    options = ImageOptions(algorithm = DitheringAlgorithm.DIFFUSION)
)
```

19. drawBox

描述: 绘制方框。

方法签名：

```
suspend fun drawBox(  
    x: Int,  
    y: Int,  
    width: Int,  
    height: Int,  
    thickness: Int  
): Result<Unit>
```

参数：

参数名	类型	必填	说明
x	Int	是	左上角X坐标（点）
y	Int	是	左上角Y坐标（点）
width	Int	是	方框宽度（点）
height	Int	是	方框高度（点）
thickness	Int	是	线宽（点）

示例代码：

```
printer.drawBox(x = 10, y = 10, width = 300, height = 200, thickness = 2)
```

20. drawBar

描述：绘制实心条。

方法签名：

```
suspend fun drawBar(  
    x: Int,  
    y: Int,  
    width: Int,  
    height: Int  
): Result<Unit>
```

参数：

参数名	类型	必填	说明
x	Int	是	左上角X坐标（点）
y	Int	是	左上角Y坐标（点）
width	Int	是	宽度（点）
height	Int	是	高度（点）

示例代码：

```
printer.drawBar(x = 10, y = 100, width = 300, height = 5)
```

21. drawCircle

描述：绘制圆。

方法签名：

```
suspend fun drawCircle(  
    x: Int,  
    y: Int,  
    diameter: Int,  
    thickness: Int  
): Result<Unit>
```

参数：

参数名	类型	必填	说明
x	Int	是	圆心X坐标（点）
y	Int	是	圆心Y坐标（点）
diameter	Int	是	直径（点）
thickness	Int	是	线宽（点）

示例代码：

```
printer.drawCircle(x = 150, y = 150, diameter = 100, thickness = 2)
```

22. drawEllipse

描述：绘制椭圆。

方法签名：

```
suspend fun drawEllipse(  
    x: Int,  
    y: Int,  
    width: Int,  
    height: Int,  
    thickness: Int  
): Result<Unit>
```

参数：

参数名	类型	必填	说明
x	Int	是	左上角X坐标（点）
y	Int	是	左上角Y坐标（点）
width	Int	是	椭圆宽度（点）
height	Int	是	椭圆高度（点）
thickness	Int	是	线宽（点）

示例代码：

```
printer.drawEllipse(x = 50, y = 50, width = 200, height = 100, thickness = 2)
```

23. erase

描述：擦除区域。

方法签名：

```
suspend fun erase(  
    x: Int,  
    y: Int,  
    width: Int,  
    height: Int  
): Result<Unit>
```

参数：

参数名	类型	必填	说明
x	Int	是	左上角X坐标（点）
y	Int	是	左上角Y坐标（点）
width	Int	是	宽度（点）
height	Int	是	高度（点）

示例代码：

```
printer.erase(x = 0, y = 0, width = 100, height = 50)
```

24. reverse

描述：反转区域。

方法签名：

```
suspend fun reverse(x: Int, y: Int, width: Int, height: Int): Result<Unit>
```

参数：

参数名	类型	必填	说明
x	Int	是	左上角X坐标（点）
y	Int	是	左上角Y坐标（点）
width	Int	是	宽度（点）
height	Int	是	高度（点）

示例代码：

```
printer.reverse(x = 0, y = 0, width = 100, height = 50)
```

25. printPDF417

描述：打印 PDF417 二维码。

方法签名：

```
suspend fun printPDF417(  
    x: Int,  
    y: Int,  
    width: Int,  
    height: Int,  
    rotate: Int,  
    options: List<String> = emptyList(),  
    expression: String  
) : Result<Unit>
```

参数：

参数名	类型	必填	默认值	说明
x	Int	是	-	X坐标（点）
y	Int	是	-	Y坐标（点）
width	Int	是	-	宽度（点）
height	Int	是	-	高度（点）
rotate	Int	是	-	旋转角度（0/90/180/270）
options	List	否	emptyList()	选项列表，每个元素格式为 'key=value'
expression	String	是	-	表达式

示例代码：

```
printer.printPDF417(  
    x = 50,  
    y = 200,  
    width = 100,  
    height = 50,  
    rotate = 0,  
    expression = "PDF417 Data"  
)
```

26. printDataMatrix

描述：打印 DataMatrix 二维码。

方法签名：

```
suspend fun printDataMatrix(
    x: Int,
    y: Int,
    width: Int,
    height: Int,
    xm: Int,
    row: Int,
    col: Int,
    expression: String
): Result<Unit>
```

参数：

参数名	类型	必填	说明
x	Int	是	X坐标（点）
y	Int	是	Y坐标（点）
width	Int	是	宽度（点）
height	Int	是	高度（点）
xm	Int	是	X模块数（1-16）
row	Int	是	行数
col	Int	是	列数
expression	String	是	表达式

示例代码：

```
printer.printDataMatrix(
    x = 50,
    y = 200,
    width = 100,
    height = 100,
    xm = 10,
    row = 10,
    col = 10,
    expression = "DataMatrix Data"
)
```

27. printMaxiCode

描述：打印 MaxiCode 二维码。

方法签名：

```
suspend fun printMaxiCode(
    x: Int,
    y: Int,
    mode: Int,
    classType: Int,
    country: Int,
    post: String,
    lm: String,
    message: String
): Result<Unit>
```

参数：

参数名	类型	必填	说明
x	Int	是	X坐标（点）
y	Int	是	Y坐标（点）
mode	Int	是	模式（2或3）
classType	Int	是	类型（0-255）
country	Int	是	国家代码
post	String	是	邮政编码
lm	String	是	位置信息（如邮编等信息）
message	String	是	消息内容

示例代码：

```
printer.printMaxiCode(
    x = 100,
    y = 100,
    mode = 2,
    classType = 0,
    country = 1,
    post = "12345",
    lm = "",
    message = "MaxiCode Data"
)
```

28. putPcx

描述：放置PCX文件。

方法签名：

```
suspend fun putPcx(x: Int, y: Int, filename: String): Result<Unit>
```

参数：

参数名	类型	必填	说明
x	Int	是	X坐标（点）
y	Int	是	Y坐标（点）
filename	String	是	文件名

示例代码：

```
printer.putPcx(x = 50, y = 50, filename = "logo.pcx")
```

29. printWaterMark

描述：打印水印。

方法签名：

```
suspend fun printWaterMark(density: Int): Result<Unit>
```

参数：

参数名	类型	必填	说明
density	Int	是	浓度（0-10，0为最浅，10为最深）

示例代码：

```
printer.printWaterMark(5)
```

30. print

描述：执行打印。标签打印的前提条件，必须在所有打印内容之后调用。

方法签名：

```
suspend fun print(copies: Int = 1): Result<Unit>
```

参数：

参数名	类型	必填	默认值	说明
copies	Int	否	1	打印份数

返回值: `Result<Unit>` 打印结果

示例代码:

```
printer.setSize(60, 40)
printer.setGap(2, 0)
printer.printText(50, 30, "Label Content")
printer.printQRCode(200, 150, "QR Data")
printer.print(copies = 1)
```

31. clear

描述: 清除缓冲区。

方法签名:

```
suspend fun clear(): Result<Unit>
```

返回值: `Result<Unit>` 操作结果

示例代码:

```
printer.clear()
```

32. feedPaper

描述: 走纸。

方法签名:

```
suspend fun feedPaper(lines: Int): Result<Unit>
```

参数:

参数名	类型	必填	说明
lines	Int	是	走纸行数

示例代码:

```
printer.feedPaper(20)
```

33. backFeed

描述：回纸。

方法签名：

```
suspend fun backFeed(len: Int): Result<Unit>
```

参数：

参数名	类型	必填	说明
len	Int	是	回纸长度（mm）

示例代码：

```
printer.backFeed(10)
```

34. limitFeed

描述：限制走纸。

方法签名：

```
suspend fun limitFeed(maxLen: Int): Result<Unit>
```

参数：

参数名	类型	必填	说明
maxLen	Int	是	最大长度（mm）

示例代码：

```
printer.limitFeed(100)
```

35. formFeed

描述：走纸到撕纸位。

方法签名：

```
suspend fun formFeed(): Result<Unit>
```

返回值: `Result<Unit>` 操作结果

示例代码:

```
printer.formFeed()
```

36. home

描述: 回到起始位置。

方法签名:

```
suspend fun home(): Result<Unit>
```

返回值: `Result<Unit>` 操作结果

示例代码:

```
printer.home()
```

37. cutPaper

描述: 切纸。

方法签名:

```
suspend fun cutPaper(): Result<Unit>
```

返回值: `Result<Unit>` 操作结果

示例代码:

```
printer.cutPaper()
```

38. sound

描述: 蜂鸣。

方法签名:

```
suspend fun sound(level: Int, interval: Int): Result<Unit>
```

参数：

参数名	类型	必填	说明
level	Int	是	音量（1-10）
interval	Int	是	间隔（毫秒）

示例代码：

```
printer.sound(level = 5, interval = 200)
```

39. selfTest

描述：执行自检。

方法签名：

```
suspend fun selfTest(): Result<Unit>
```

返回值：`Result<Unit>` 操作结果

示例代码：

```
printer.selfTest()
```

40. 附录

40.1 字体枚举 (TsplFont)

值	说明
FONT_0	字体0 (8x12)
FONT_1	字体1 (12x16)
FONT_2	字体2 (16x24)
FONT_3	字体3 (24x32)
FONT_4	字体4 (32x48)
FONT_5	字体5 (14x19)
FONT_6	字体6 (21x27)
FONT_7	字体7 (28x36)
FONT_8	字体8 (56x72)
FONT_9	字体9 (21x27)
FONT_10	字体10 (16x20)
FONT_ROMAN	罗马字体
FONT_SANS	无衬线字体

40.2 旋转角度枚举 (Rotation)

值	说明
NORMAL	正常 (0度)
ROTATE_90	旋转90度
ROTATE_180	旋转180度
ROTATE_270	旋转270度

40.3 条码类型枚举 (BarcodeType)

值	说明
CODE128	Code 128
CODE128A	Code 128 A
CODE128B	Code 128 B
CODE128C	Code 128 C
CODE39	Code 39
CODE93	Code 93
EAN13	EAN-13
EAN8	EAN-8
UPC_A	UPC-A
UPC_E	UPC-E
CODABAR	Codabar
ITF	ITF

40.4 抖动算法枚举 (DitheringAlgorithm)

值	说明
THRESHOLD	简单阈值
DIFFUSION	扩散抖动
CLUSTER	聚集抖动

40.5 错误类型说明

错误类型	说明	可能的触发场景
ConnectionError	连接错误	打印机连接失败、网络连接问题、蓝牙配对失败
CommunicationError	通信错误	数据传输失败、打印机响应异常、通信超时
InvalidParameterError	参数无效错误	传入的参数值超出范围、参数格式不正确
TimeoutError	超时错误	操作执行超时、打印机无响应
NotConnectedError	未连接错误	尝试在未连接状态下执行操作
UnsupportedOperationError	不支持的操作错误	打印机型号不支持的功能、协议不支持的命令
UnknownError	未知错误	其他未分类的错误情况

错误码说明：接口返回 `Result.Failure` 时，`PrinterError` 携带 `errorCode` 字段（对应 `UXPErrCode.value`）。每个错误类型默认取下表兜底错误码，SDK 各调用点会传入更具体的错误码；完整错误码表见《UXPErrCode 错误码说明》（Guide 目录）。

兜底错误码	数值	说明
ConnectionError	1001	连接错误
CommunicationError	1002	通讯错误
InvalidParameterError	1003	参数无效
TimeoutError	1004	超时
NotConnectedError	1005	未连接
UnsupportedOperationError	1006	不支持的操作
UnknownError	1007	未知错误

错误处理示例：

```
val result = printer.printText("Hello, World!")
when (result) {
    is Result.Success -> {
        Log.d("Printer", "Print successful")
    }
    is Result.Failure -> {
        when (result.error) {
            is PrinterError.ConnectionError -> {
                Log.e("Printer", "Connection error: ${result.error.message}")
            }
        }
    }
}
```

```
        is PrinterError.CommunicationError -> {
            Log.e("Printer", "Communication error:
${result.error.message}")
        }
        is PrinterError.InvalidParameterError -> {
            Log.e("Printer", "Invalid parameter: ${result.error.message}")
        }
        is PrinterError.TimeoutError -> {
            Log.e("Printer", "Timeout error: ${result.error.message}")
        }
        is PrinterError.NotConnectedError -> {
            Log.e("Printer", "Not connected: ${result.error.message}")
        }
        is PrinterError.UnsupportedOperationError -> {
            Log.e("Printer", "Unsupported operation:
${result.error.message}")
        }
        is PrinterError.UnknownError -> {
            Log.e("Printer", "Unknown error: ${result.error.message}")
        }
    }
}
}
```