EPSON
EXCEED YOUR VISION

Advanced Printer Driver for TM-T81 Ver.4

Status API 手册

M00024205-3
Rev. F

注意事项

- 未经 Seiko Epson Corporation 事先书面同意，不得翻印、在检索系统中存储或以任何形式或通过任何方式（电子、机械、影印、录制等）传送本文档的任何部分。
- 本文档的内容如有变更，恕不另行通知。请联系我们以获取最新信息。
- 在准备本文档的阶段虽尽了最大努力，但错误或疏漏在所难免， Seiko Epson Corporation 不对此负责。
- 对于使用本文档中包含的信息而引起的损失 , 我们也将不负任何责任。
- Seiko Epson Corporation 及其联营公司不对本产品的购买者或第三方因以下问题而造成的损坏、损失、费用或开支负责：事故、误用或滥用本产品，或未经授权修改、修理或改变本产品，或（不包括美国）不能严格按照 Seiko Epson Corporation 的操作和维护指示进行操作。
- 除了由Seiko Epson Corporation指定为原装EPSON产品或EPSON认可产品的以外，Seiko Epson Corporation不负责因使用任何其他选件或易耗件产品而导致的任何损坏或问题。

商标

EPSON® 和 ESC/POS® 为 Seiko Epson Corporation 在美国和其他国家的注册商标。

MS-DOS®、 Microsoft®、 Win32®、 Windows®、 Windows Vista®、 Windows Server®、 Visual Studio®、 Visual Basic®、 Visual C++® 和 Visual C#® 为 Microsoft Corporation 在美国和其他国家的注册商标或商标。

ESC/POS® 指令系统

EPSON 通过其自有的 POS 打印机命令系统 (ESC/POS) 获取了业界主动性。ESC/POS 拥有大量包含专利技术的命令。其高可扩展性使用户可以构建多用途多功能的 POS 系统。该系统兼容除 TM-C100 之外的所有类型的 EPSON POS 打印机和显示设备。而且其灵活性便于今后升级系统。其功能性和易用性被全世界所重视。

安全须知

标记事项

本手册中的标记按如下定义的重要性级别进行标识。在使用本产品之前 , 请仔细阅读以下信息。

CAUTION	提供必须遵守以避免设备损坏或故障的信息。
NOTE	提供重要信息和实用提示。

使用限制

将本产品用于需要高度可靠性 / 安全性的应用时，例如与航空、铁路、海运、汽车等相关的运输设备、防灾设备、各种安全设备、或功能性 / 精密设备等，您应当在考虑将故障保险和冗余机制加入设计中以维持安全和整体系统可靠性之后再使用本产品。因为本产品不设计为被应用于需要极高可靠性 / 安全性的应用，例如航空设备、主要通讯设备、核电控制设备或与直接医疗相关的医学设备，请在进行完全评估之后自行判断是否适用本产品。

关于本手册

手册的用途

本手册的目的是使用 Status API 为开发和设计打印机应用程序提供所有必要的信息。

手册内容

本手册包括以下部分：

目录

概述

Status API （状态应用程序接口）用于监测 Epson TM 打印机的 API 状态。监测 TM 打印机的高级功能可以嵌入具有打印功能的应用程序中。

手册结构

安装手册

描述了从安装 APD 到执行打印测试，添加打印机驱动程序与自动 APD 静态安装的过程。

TM 打印机手册

描述了如何使用 APD 及其功能。
描述了 TM-T81 的规格。

Status API 手册

描述了如何使用 Status API 从用户应用程序中获得 TM 打印机的状态。

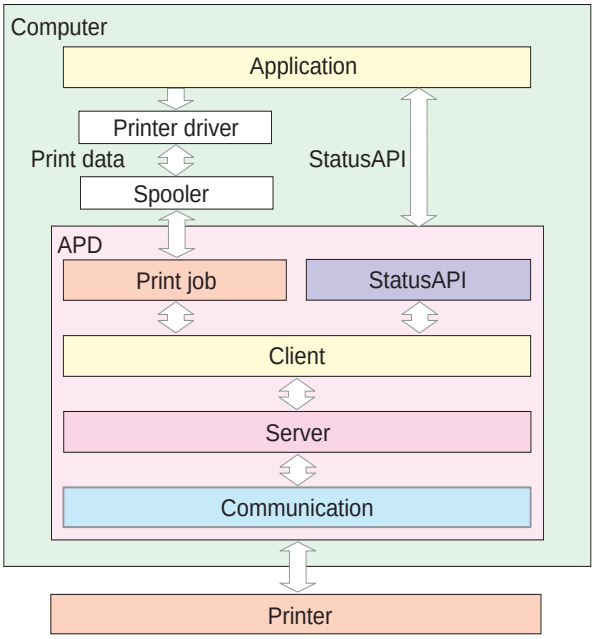
Devmode API / PRINTERINFO 手册

描述了如何使用 Devmode API 在用户应用程序中配置打印机的一些功能。
描述了 Windows 环境下的 PRINTERINFO 功能。

Status API 概要

Status API 系统

Status API 接收从 TM 打印机发出的信号，并且始终保留打印机最新的状态。当需要时，应用程序可以获取最新的信息。



CAUTION 在 Terminal Service / Citrix XenApp 环境下， Status API 不能被使用。

术语表

术语	解释说明
APD	Advanced Printer Driver：TM 打印机的 Windows 打印机驱动程序。不同于普通的 Windows 打印机驱动， Status API 是用来监测打印机状态而被同时安装的。
ASB 状态	Auto Status Back：TM 打印机的功能之一。当打印机状态发生改变时 （打开或者关闭盖板，缺纸，打印结束等）此状态将被自动发出。
维护计数器	用于记录 TM 打印机运行状态的内部计数器。例如自动切纸器的运行次数，打印机运行时等。

可以从 TM 打印机获取的信息

术语	解释说明
ASB 状态	打印应用程序所需的信息，例如，打印结束，脱机，缺纸，盖板打开，电源关闭，错误发生等。这些信息被自动发送至 Status API。
维护计数器	获取信息，例如进纸行数，自动切纸器运行次数，运行时等。 用于打印机管理应用程序。存在可以从 Status API 重置的计数器和不能被重置的整体计数器。

开发语言

Win32

- Visual Basic 6.0
- Visual C++

.NET

- Visual Basic .NET
- Visual C#

> **CAUTION**
>
> .NET Framework 版本
>
> 确认 APD 环境。请参照 “安装手册”。
>
> 如果您想使用 Windows XP 下的 Status API .NET Wrapper 时，请在安装 APD 前先安装 .NET Frame Work 2.0 或更高版本。

使用 Status API

本章解释说明使用 Status API 的应用程序开发环境的结构，获取 ASB 状态的方法和获取 ASB 状态的步骤。有关其它功能，请参照第 27 页 “Win32 参考”

安装和卸载

在 APD(Advanced Printer Driver) 安装 / 卸载时 Status API 也被安装 / 卸载。有关详情，请参照“安装手册”。

开发环境的结构

使用 Status API 的应用程序开发环境的结构根据开发工具而异。

> **CAUTION**
> 本手册的用户界面是在英语语言环境下进行说明的。
> 其它的语言请在相应的语言环境下阅读参考。

Visual Basic

以下是采用 Visual Basic 构建的开发环境的示例。

1. 在安装示例程序的文件夹中复制 StatusAPI.bas(默认路径是 C:\Program Files\EPSON\EPSON Advanced Printer Driver 4 \Sample\US\Src\VB6\SingleFunction\Program09"），在开发应用程序时将其粘贴到运行文件夹中。
2. 启动 Microsoft Visual Basic 并打开项目界面。
3. 从菜单栏的 [Project] 中选择 [Add a standard module]。
4. Add standard module 界面出现。选择 [Existing file] 选项卡，并指定步骤 1 中从示例程序复制的 “StatusAPI.bas”。点击 [Open] 按钮。
   StatusAPI.bas”被加入项目资源管理器。
5. 从菜单栏的 [Project] 中选择 [Reference settings]。
6. Reference settings 界面出现。通过 [Reference library file] 中的 “Microsoft DAO 3.6 Object Library”放置一个复选框，并点击 [OK]。

C++

以下是使用 C++ 构建的开发环境的示例。

1 启动 Microsoft Visual C++ 并打开项目界面。

2 从 APD 的安装文件夹中复制 EpsStmApi.h，并粘贴到开发应用程序时使用的运行文件夹中 （项目创建的文件夹）。

3 打开源文件。使用 #include 标识定义 EpsStmApi.h。
定义方法：*#include "EpsStmApi.h"*

Visual Basic .NET

以下是使用 Visual Basic .NET 创建开发环境的示例。

1 启动 Microsoft Visual 2005 并打开 Visual Basic .NET 项目界面。

2 在 Solution Explorer 中右击 [References]，并选择 [Add References]。

NOTE 如果 [References] 项未出现，请在 Solution Explorer 中点击 [Show All Files] 图标。

3 “Add References”界面出现。点击 [Browse] 选项卡。

4 在 [Look in] 中指定 “C:\WINDOWS\assembly”。

5 按照以下命名规则键入文件名，并点击 [OK]。
“GAC_MSIL\EpsonStatusAPI\(EpsonStatusAPI 的版本)_(EpsonStatusAPI 的 Public Key Token)\EpsonStatusAPI.dll”

例如：GAC_MSIL\EpsonStatusAPI\4.0.9.0__46bb02e1480038cb\EpsonStatusAPI.dll

6 在 Solution Explorer 中选择 [References] - [EpsonStatusAPI]，并在属性中为 [Specific Version] 选择 “False”。

7 在源代码的开头使用 Imports statement，如下所述。
Imports com.epson.pos.driver

8 Visual Basic .NET 环境使用 Status API 可以开发应用程序。

Visual C#

以下是使用 Visual C# 创建开发环境的示例。

1 启动 Microsoft Visual 2005，并打开 Visual C# 项目界面。

2 在 Solution Explorer 中右击 [References]，并选择 [Add References]。

NOTE 如果 [References] 项未出现，请在 Solution Explorer 中点击 [Show All Files] 图标。

3 “Add References”界面出现。点击 [Browse] 选项卡。

4 在 [Look in] 中指定 “C:\WINDOWS\assembly”。

5 按照以下命名规则键入文件名，并点击 [OK]。
“GAC_MSIL\EpsonStatusAPI\(EpsonStatusAPI 的版本)_(EpsonStatusAPI 的 Public Key Token)\EpsonStatusAPI.dll”

例如：GAC_MSIL\EpsonStatusAPI\4.0.9.0__46bb02e1480038cb\EpsonStatusAPI.dll

6 在 Solution Explorer 中选择 [References] - [EpsonStatusAPI]，并在属性中为 [Specific Version] 选择 “False”。

7 在源代码的开头使用 using keyword，如下所述。

```
using com.epson.pos.driver
```

8 Visual C# 环境使用 Status API 可以开发应用程序。

Status API 函数的类型

Status API 有以下函数。有关函数的详情，请参照第 27 页 “Win32 参考”。对于不同的打印机机型被支持的函数也不同。有关各种机型的详情，请参照第 27 页 “Win32 参考”。

应用程序	函数	说明
开始 / 关闭 Status API	BiOpenMonPrinter	调用指定的打印机使用 Status API。
	BiCloseMonPrinter	关闭 Status API。
占用 TM 打印机	BiLockPrinter	占用 TM 打印机。占用作为共享打印机使用的 TM 打印机。 在占用期间，打印机不接收其他进程的 API。
	BiUnLockPrinter	取消 BiLockPrinter。
获取 ASB 状态	BiGetStatus	当应用程序获取时从 Status API 获取 ASB 状态。
	BiSetStatusBackFunction	提供有关调用回调函数通知应用程序 Status API 的 ASB 状态何时发生变更的通知。
	BiSetStatusBackFunctionEx	提供有关调用回调函数通知应用程序 Status API 的 ASB 状态何时发生变更的通知。 同时获取端口号。
	BiSetStatusBackWnd	当 Status API 的 ASB 状态改变时触发一个按钮点击事件。
	BiCancelStatusBack	退出自动状态通知功能。此功能可用于 BiSetStatusBackFunction，BiSetStatusBackFunctionEx 和 BiSetStatusBackWnd。
获取和重置维护计数器	BiGetCounter	获取打印机维护计数器的值。
	BiResetCounter	重置打印机维护计数器。
获取打印机信息	BiGetType	获取 BM 传感器和客户显示连接状态等 TM 打印机信息。
	BiGetPrnCapability	获取打印机的信息，例如 firmware 等。
画笔控制	BiOpenDrawer	打开货币纸盒。
恢复可能恢复的错误	BiCancelError	当错误状态 （例如卡纸等）被清除后，使用此函数使打印机自动裁纸器从错误状态中恢复。无需重启，恢复打印机至待机状态。
打印机重置	BiResetPrinter	重置并行 /USB/Ethernet 接口打印机。无法重置串行接口打印机。
	BiForceResetPrinter	也可以重置被 BiLockPrinter 占用的 TM 打印机。

应用程序	函数	说明
电源关闭预处理	BiPowerOff	设置电源关闭或待机模式。 进行以下操作： • 存储维护计数器的值。 • 将界面设为 BUSY。 • 将打印机设为待机模式。
命令定义文件	BiSendDataFile	分别对创建到打印机的命令定义文件进行定义。该命令不执行。
	BiDirectSendRead	发送已定义的命令定义文件的命令到打印机并执行命令。
发送 ESC/POS 命令	BiDirectIO	可以传输 ESC/POS 命令到打印机并从打印机接收数据。
	BiDirectIOEx	可以发送和接收 ESC/POS 命令。不添加 ASB 禁止命令。

获取 ASB 状态

从应用程序中获取 ASB 状态的方法和函数如下所示。

使用场合	Status API
应用程序需要时获取	BiGetStatus
获取 ASB 状态的方式如下	BiSetStatusBackFunction
	BiSetStatusBackFunctionEx
	BiSetStatusBackWnd

获取 ASB 状态的方式如下。

- 确认打印机是否可以预先打印。
- 确认打印成功结束。通过宏定义的 “ASB_PRINT_SUCCESS”（常量）来确认。
- ASB 状态监测主打印机的状态，例如，缺纸、盖板打开、打印机连接状态等。所以推荐即使不打印时，请持续监测打印机。

有关获取的 ASB 状态，请参照第 22 页 “ASB 状态”

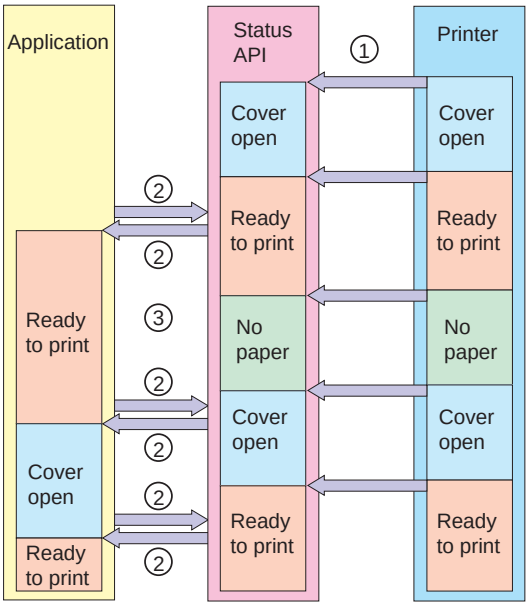
NOTE

必须使用 BiOpenMonPrinter 启动 Status API，当获取 ASB 状态时，打印机必须开启。

BiGetStatus

用户 （或应用程序）需要时， BiGetStatus 可获取 ASB 状态。

例如：以下图表解释说明了 Status API 的流程以及调用 BiGetStatus 时 ASB 状态的流程。



[1] 每当打印机状态改变时，打印机会使用 ASB 功能自动发送 ASB 状态到 Status API。 Status API 存储最新的 ASB 状态。

[2] 当 ASB 状态需要时应用程序调用 BiGetStatus。Status API 发送存储的 ASB 状态到应用程序。

[3] 当应用程序没有请求时，即使打印机的 ASB 状态改变， Status API 也不会发送 ASB 状态。

有关 BiGetStatus 的语法，请参照第 46 页 “BiGetStatus”。

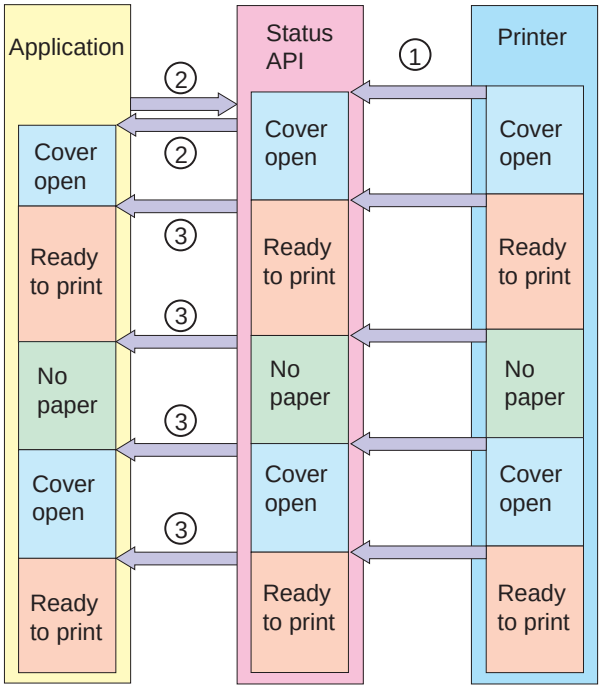
BiSetStatusBackFunction

BiSetStatusBackFunction 是 API 的函数之一，可使用回调函数自动允许应用程序获取最新的 ASB 状态。
使用回调函数调用 BiCancelStatusBack 从 Status API 中取消 ASB 状态通知。有关详情，请参照第 51 页 “BiCancelStatusBack”。

> **CAUTION** 当开发环境是 Visual Basic 时此项不可用。

> **NOTE** BiSetStatusBackFunctionEx 可以识别回调来自哪台打印机，同时还能识别 BiSetStatusBackFunction 函数。

例如：以下图表解释说明了 Status API 的流程以及调用 BiSetStatusBackFunction 时 ASB 状态的流程。



[1] 每当打印机状态改变时，打印机会使用 ASB 功能自动发送 ASB 状态到 Status API。

[2] BiSetStatusBuckFunction 注册回调函数。

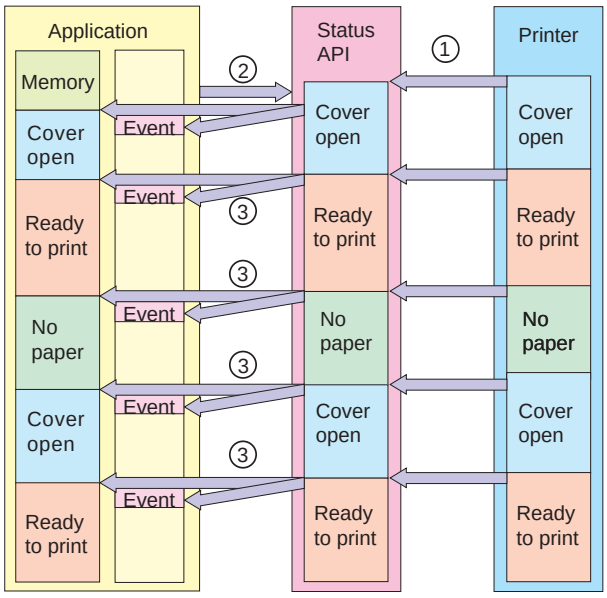
[3] 每当打印机的 ASB 状态改变时，Status API 会调用回调函数并通知应用程序最新的 ASB 状态。

有关 BiSetStatusBackFunction 的语法，请参照第 48 页 “BiSetStatusBackFunction”。

BiSetStatusBackWnd

BiSetStatusBackWnd 是 API 的函数之一，它通过注册应用程序界面按钮和内存地址存储状态的 Window 句柄来获取最新的 ASB 状态。
使用按钮点击事件调用 BiCancelStatusBack 从 Status API 中取消 ASB 状态通知。有关详情，请参照第 51 页 “BiCancelStatusBack”。

例如：以下图表解释说明了 Status API 的流程以及调用 BiSetStatusBackWnd 时 ASB 状态的流程。



[1] 每当打印机状态改变时，打印机会使用 ASB 功能自动发送 ASB 状态到 Status API。Status API 存储最新的 ASB 状态。

[2] 当 BiSetStatusBackWnd 注册应用程序界面按钮和内存地址存储状态的 Window 句柄时， Status API 设置数据到指定的地址并发送按钮点击事件。

[3] 每当打印机的 ASB 状态改变时，Status API 会发送打印机最新的 ASB 状态到指定的内存，并发送一个按钮点击事件。

有关 BiSetStatusBackWnd 的语法，请参照第 50 页 “BiSetStatusBackWnd”

Status API 错误及对应方案

Status API 错误是指通知 ASB 状态和调用 Status API 时发生的错误。下文解释说明了其错误和对应方案的详细内容。请参照下文解决应用程序错误。

ASB 状态

下表所示是获取 ASB 状态时返回的错误。针对不同机型内容可能存在差异。有关详情，请参照第 87 页 “机型信息”。

宏定义（常量）	原因	对应方案
ASB_NO_RESPONSE	打印机电源未开启。 通讯电缆未连接。 指定的打印机名称 / 端口与不同。	确认打印机的状态和端口，例如，电缆等。
ASB_PRINT_SUCCESS	通报打印成功结束。不通知其它情况。	-
ASB_DRAWER_KICK	货币纸盒打开。	刻意打开货币纸盒时没有问题。
ASB_OFF_LINE	发生导致打印机脱机的错误。	排除导致打印机脱机的原因。
ASB_COVER_OPEN	盖板打开。	关闭打印机盖板。
ASB_PAPER_FEED	进纸。	进纸时没有问题。
ASB_AUTOCUTTER_ERR	发生了一个自动裁纸器错误。	排除错误的原因并重启打印机或者发送一个错误 恢复命令 （BiCancelError）。 *
ASB_UNRECOVER_ERR	打印机发生打印错误。	立即关闭打印机电源。 *
ASB_AUTORECOVER_ERR	打印头温度上升。	如果打印头温度逐渐降低，错误会自动取消。 *
ASB_RECEIPT_NEAR_END	打印纸将尽。	更换打印纸。
ASB_RECEIPT_END	缺纸。	更换打印纸。
ASB_VALIDATION_NO_PAPER	验证检测器未检测到打印纸。	-

* 详细说明请参照各打印机的操作手册。

NOTE

当创建开发环境时，使用 EPSStmApi.h 或者 StatusAPI.bas 文件进行宏定义。

Status API 执行错误

调用 Status API 函数时，发生下表所列错误。针对不同的 Status API 函数内容存在差异。

宏定义 （常量）	原因	对应方案
ERR_TYPE	nType 参数不同。	指定正确的值。
ERR_OPENED	指定的打印机已经开启。	因为打印机已经开启，使用其句柄值或指定其它打印机。
ERR_NO_PRINTER	指定的打印机驱动程序不存在。	确认打印机驱动程序名称。
ERR_NO_TARGET	找不到指定的打印机。 连接了一台未指定的打印机。	连接到正确的打印机。
ERR_NO_MEMORY	没有足够的内存空间。	增加可用的内存空间。
ERR_HANDLE	打印机指定的句柄值不正确。	确认句柄值。
ERR_TIMEOUT	超时错误。	如果错误持续发生，确认打印机是否正确连接。
ERR_ACCESS	打印机无法执行读 / 写操作。 （打印机电源未打开或电缆未正确连接等。）	确认打印机。 （打印机电源，电缆连接等。）
ERR_PARAM	参数错误。	由于指定的参数不正确，请仔细阅读语法。
ERR_NOT_SUPPORT	不被支持的机型。	不能使用不被支持的机型。
ERR_EXIST	指定的数据已经存在。	删除已经存在的数据。 例如： 当执行 BiSetStatusBackXXX 时发生此错误，请在执行 BiCancelStatusBack 之后重新尝试。
ERR_EXEC_FUNCTION	Status API 正被其它应用程序使用，所以此函数不可用。	关闭被其它应用程序使用的 Status API。
ERR_PH_NOT_EXIST	PortHandler 不运行，或 PortHandler 用户端和服务器之间存在通讯错误。	检查 PortHandler 用户端和服务器之间的通讯，然后重启计算机。
ERR_SPL_NOT_EXIST	卷轴停止工作。	确认 Print Spooler 是否正常工作。 （控制面板 - 管理工具 - 服务）
ERR_RESET	当打印机正在被重置时此函数不可用。	等待后重新调用。
ERR_LOCKED	打印机被锁住。	等待直到打印机变为非锁定状态，或在锁定该打印机的程序中执行 BiUnlockPrinter。

NOTE　当创建开发环境时，使用 EPSStmApi.h 或者 StatusAPI.bas 文件进行宏定义。

如何使用共享打印机

当在开发应用程序过程中使用共享打印机时，请注意以下内容。

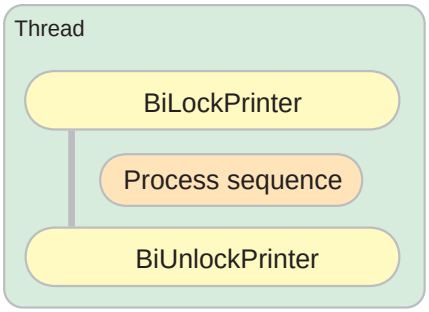
- 设备在整个使用过程中必须是唯一访问的。
- 必须处理 BiLockPrinter 错误。
- 尽量缩短独享打印机访问的时间。

> **CAUTION** 当设置防火墙时，请将端口号 2291 设置为 [Exception]

创建独享访问

对进程序列的唯一控制

创建一个应用程序可在 BiLockPrinter 和 BiUnlockPrinter 之间放入每一个进程序列。



> **NOTE** 当您在应用程序中只引用了一个 API 时，访问不需要独享。

BiLockPrinter 错误处理

当打印机从另一个进程被访问时， BiLockPrinter 会返回一个错误 （ERR_LOCKED）。创建一个应用程序以便处理该错误，然后再次执行 BiLockPrinter。

缩短独享打印机访问的时间

当打印机被独享访问时，其它进程不能执行唯一的 API 且打印禁用。
因此需要尽量缩短独享访问的时间来改善系统性能。

程序示例

```
int nRet = BiLockPrinter(1, 1000);
if (nRet == SUCCESS) {
 // 锁定打印机以允许对下列 API 的独享访问权。结束时解锁打印机。
BiSCNSetImageFormat(1, EPS_BI_SCN_JPEGNORMAL);
BiSCNReadImage(1, 1, EPS_BI_SCN_CHECKPAPER, 0, 0, NULL, EPS_BI_SCN_NVMEMORY_NOTSAVE);
// 解锁打印机。
BiUnlockPrinter(1);
} else {
 // 独享访问失败时进行错误处理。

}
```

使用 APD3.xx 应用程序时

APD3.xx 不支持打印机独享访问，因此您的应用程序如果支持 APD3.xx，则没有独享访问的功能。因此，当您仅使用既存的应用程序时，无需修改应用程序即可获得打印机的独享访问。

如果想从其它进程进行访问，修改应用程序来建立独享访问，请参照第 24 页 “创建独享访问”。

Win32 参考

本章描述 TM-T81 所使用的 Status API 和语法。有关 TM-T81 的 ASB 状态，导致打印机维护计数器的问题，请参照第 87 页 “机型信息”。

NOTE 数据类型在 C++ 中进行描述。

TM-T81 所使用的 Status API

Status API	页数	Status API	页数
BiOpenMonPrinter	28	BiCloseMonPrinter	30
BiLockPrinter	31	BiUnlockPrinter	33
BiSetMonInterval	34	BiSetMonEtherInterval	35
BiDirectIO	36	BiDirectIOEx	38
BiResetPrinter	41	BiForceResetPrinter	43
BiCancelError	44	BiGetType	45
BiGetStatus	46	BiGetRealStatus	47
BiSetStatusBackFunction	48	BiSetStatusBackFunctionEx	49
BiSetStatusBackWnd	50	BiCancelStatusBack	51
BiPowerOff	52	BiGetCounter	53
BiResetCounter	55	BiGetPrnCapability	57
BiOpenDrawer	58	BiSendDataFile	60
BiDirectSendRead	62	BiSetDefaultEchoTime	65
BiSetEtherEchoTime	66	BiSetReadWaitTimeOut	67

BiOpenMonPrinter

使 Status API 可用并返回其句柄。
可打开一台打印机同时进行多个进程。
当从同一进程再次打开一台已开启的打印机时，会返回一个新的句柄。在这种情况下，新旧两个句柄均有效。

语法

nHandle = **BiOpenMonPrinter** *(INT nType, LPSTR pName)*

例如）

- 从端口中使 Status API 可用。
  nHandle = BiOpenMonPrinter(1,"ESDPRT001");
- 从打印机中使 Status API 可用。
  nHandle = BiOpenMonPrinter(2, "EPSON TM-T88IV Receipt");

变量

nType: 指定 pName 类型。此为 INT 类型。

宏定义 （常量）	值	说明
TYPE_PORT	1	指定 *pName* 中的端口名称。
TYPE_PRINTER	2	指定 *pName* 中的打印机名称。

pName: 如果在 nType 中 1 被指定，则指定端口名称 （例如：“ESDPRT001”）。
如果 2 被指定，则指定打印机名称 （例如：“EPSON TM-T88IV Receipt”）。
此为 LPSTR 类型。

返回值

返回在 INT 类型中定义的变量 （*nHandle*）。如果 Status API 被成功使用，则标识打印机的句柄被返回至 *nHandle* （正确的值）。即使打印机脱机句柄也被返回。
下表返回 Status API 执行错误 （值）。

宏定义 （常量）	值	说明
ERR_TYPE	-10	*nType* 参数错误
ERR_OPENED	-20	指定的打印机已经开启。
ERR_NO_PRINTER	-30	指定的打印机驱动程序不存在。
ERR_NO_TARGET	-40	打印机不可用。
ERR_NO_MEMORY	-50	没有足够的内存空间
ERR_TIMEOUT	-70	超时错误
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_PARAM	-90	参数错误
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_SPL_NOT_EXIST	-350	卷轴停止工作。

NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

在使用其它 Status API 函数前调用此函数。返回值的句柄作为变量被其它 Status API 函数使用。

NOTE 当此函数被调用时，在调用函数 BiCloseMonPrinter 前指定的打印机被独享。在此过程中，其它应用程序中的 Status API 函数不可用。
获取的句柄仅在相同的应用程序中有效。
可同时开启的最大打印机数为 32。

调用此函数时，根据打印机的状态执行如下表所示的操作。

打印机状态	操作
联机	返回句柄至 *nHandle*。
脱机	返回句柄至 *nHandle*。但是必须切换到联机状态，因为打印机无法脱机打印。
电缆移除 / 电源关闭	返回 “ERR_ACCESS” 至 *nHandle*。

BiCloseMonPrinter

退出对打印机的监测状态。

> NOTE 当 BiOpenMonPrinter 函数被调用时，始终使用 BiCloseMonPrinter 函数退出对打印机的监测状态。如果 BiOpenMonPrinter 未退出而再次被调用，则会发生错误。

语法

nErr = **BiCloseMonPrinter** (*nHandle*)

变量

nHandle: 指定句柄。此为 INT 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。

> NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

BiLockPrinter

锁定打印机。

NOTE

- 本 API 用于共享打印机。
- 当使用本地打印机时，本 API 用于控制多个进程。

语法

nErr = **BiLockPrinter** (*nHandle, timeout*)

变量

nHandle: 指定句柄。此为 INT 类型。
timeout: 指定超时时间，单位 ms （毫秒）。用决定值指定。此为 DWORD 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_EXEC_FUNCTION	-310	Status API 正被其它应用程序使用，所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_RESET	-400	打印机正在重启，无法调用
ERR_LOCKED	-1000	打印机被锁定。

NOTE

有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

本 API 允许您独享访问 TM 打印机。BiUnlockPrinter API 用于退出独享访问。当 TM 打印机被独享访问时，打印机拒绝其它 API 访问的请求。打印机将返回 ERR_LOCKED 至其它 API 请求。

TM 打印机的独享访问权被给予进程。因此，在锁定 TM 打印机的同一进程中，不同线程独享 API 访问可用。

在同一进程中 API 可以被反复执行。在这种情况下，打印机被多个访问锁定。要解锁打印机，执行 BiUnlockPrinter 的次数与执行 API 的次数相同。

当从客户端对一台共享打印机或者通过 Ethernet 对一台本地打印机进行独享访问时，如果连接丢失则访问被中断，但可以通过重新连接来恢复访问。

然而，当独享访问状态被中断时，其它进程可锁定 TM 打印机从而获得其独享访问。一旦打印机被其它进程锁定，则打印机会返回 ERR_LOCKED 至先前进程的 API。 当另一进程结束解锁打印机，则先前进程的独享访问状态恢复。

以下是可能引起连接失败的原因。

[对通过 Ethernet 连接的打印机进行独享访问失败]

- 打印机关闭，或计算机与打印机之间的 Ethernet 连接断开。
- 计算机进入待机或休眠模式。

[从客户端对共享打印机进行独享访问失败]

- 客户端和服务器之间的连接断开。
- 客户端计算机进入待机或休眠模式。

BiUnlockPrinter

解锁打印机。

NOTE

- 本 API 用于共享打印机。
- 当使用本地打印机时，本 API 用于控制多个进程。

语法

nErr = **BiUnlockPrinter** (*nHandle*)

变量

nHandle: 指定句柄。此为 INT 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_EXEC_FUNCTION	-310	Status API 正被其它应用程序使用，所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_RESET	-400	打印机正在重启，无法调用
ERR_LOCKED	-1000	打印机被锁定。

NOTE

有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

本 API 解锁被 “BiLockPrinter”锁定的打印机。解锁后，打印机可接收其它进程的 API。

如果在打印机未被锁定时执行本 API，则 “SUCCESS”将被返回至*返回值*。

BiSetMonInterval

配置 Status API 读取打印机状态的时间间隔。

> **CAUTION** 时间间隔设置过长则会超出连续接收缓冲区的范围，且无法获取准确的 ASB 状态。

语法

nErr = **BiSetMonInterval** (*nHandle, wNoPrnInterval, wPrnInterval*)

变量

nHandle: 指定句柄。此为 INT 类型。
wNoPrnInterval: 未使用。
wPrnInterval: 指定时间间隔以毫秒为单位检测 Status API 的状态。此为 WORD 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_PARAM	-90	参数错误
ERR_EXEC_FUNCTION	-310	Status API 正被其它应用程序使用，所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_LOCKED	-1000	打印机被锁定。

> **NOTE** 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

如果 API 未指定检测 Status API 状态的时间间隔，则默认值为 100 毫秒。

> **CAUTION**
> - 有关支持Ethernet的接口，该程序由BiSetMonEtherInterval执行。请参见第35页 “BiSetMonEtherInterval”。
> - 执行本 API 之前，先执行 BiUnlockPrinter。

BiSetMonEtherInterval

Status API 设置网络打印机状态的读取时间间隔。

> **CAUTION** 如果您打印机的接口是并行接口，串行接口或 USB 接口，则该程序由 BiSetMonInterval 执行。请参见第 34 页 “BiSetMonInterval”。

语法

nErr = **BiSetMonEtherInterval** (*nHandle, wEtherInterval*)

变量

nHandle: 指定句柄。此为 INT 类型。

wEtherInterval: Specifies the interval to monitor the status of Status API (1 to 65) in second.
此为 WORD 类型。即使配置了大于 65 秒的值，实际将配置 65 秒。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_PARAM	-90	参数错误
ERR_NOT_SUPPORT	-100	不支持
ERR_EXEC_FUNCTION	-310	Status API 正被其它应用程序使用，所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_LOCKED	-1000	打印机被锁定。

> **NOTE** 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

如果 API 未指定检测 Status API 状态的时间间隔，则默认值为 3 秒。

> **CAUTION** 执行本 API 之前，先执行 BiUnlockPrinter。

BiDirectIO

发送特殊命令 （ESC/POS 命令）至打印机。也可从打印机获取命令执行结果。推荐使用 BiDirectIOEx 来获取执行结果。

> NOTE 有关 ESC/POS 命令请联系经销商。

语法

nErr = **BiDirectIO** (*nHandle, writeLen, writeCmd, readLen, readBuff, Timeout, nullTerminate*)

请参照下一变量。

变量

nHandle:	指定句柄。此为 INT 类型。
writeLen:	指定写入打印机的数据长度。当为 “0” 时则不写入打印机。 此为 BYTE 类型。
writeCmd:	指定写入打印机的数据 （ESC/POS 命令）。此为 LPBYTE 类型。
readLen:	指定从打印机读取的数据长度。 当打印机需要命令执行结果时指定。 当不需要时指定为 “0”。此为 LPBYTE 类型。
readBuff:	指定保存从打印机读取的数据的缓冲。 此为 LPBYTE 类型。
Timeout:	以 ms （毫秒）为单位指定超时时间。此为 DWORD 类型。
nullTerminate:	在 “True” 的情况下，当从打印机接收 NULL 时读取结束。此时，指定 readBuff 大小至 readLen。在 “FALSE” 的情况下，在 readLen 中指定的数据长度被读取，或发生超时错误前数据从打印机被读取。

> NOTE 请确保 readBuff 的大小与 readLen 中指定的长度一致或大于其值。
> 有关 ESC/POS 命令请联系经销商。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_TIMEOUT	-70	超时错误
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_PARAM	-90	参数错误
ERR_BUFFER_OVER_FLOW	-140	缓冲区空间不足
ERR_EXEC_FUNCTION	-310	Status API 正被其它应用程序使用，所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_RESET	-400	打印机正在重启，无法调用
ERR_LOCKED	-1000	打印机被锁定。

NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

通过确认 *nErr* 的返回值来确认函数是否正确执行，或通过确认打印机的运行状况来确认命令是否正确执行。如果从打印机获取了执行结果 （指定 *readLen*），确认该执行结果。
调用此函数时，根据打印机的状态执行如下表所示的操作。

打印机状态	操作
联机	返回 “SUCCESS” 至 *nErr*。 执行命令。
脱机	当在 *Timeout* 指定的时间内成功结束与打印机的通讯时，会返回 “SUCCESS”。 当在 *Timeout* 指定的时间内与打印机通讯失败时，会返回 “ERR_TIMEOUT”。
电缆移除 / 电源关闭	返回 “ERR_ACCESS” 至 *nErr*。
打印中	返回 “ERR_LOCKED” 至 *nErr*。

BiDirectIOEx

发送特殊命令 （ESC/POS 命令）至打印机。也可从打印机获取命令执行结果。与 BiDirectIO 不同， ASB 禁止命令可以被添加。当 ASB 禁止命令添加后，无法从打印机发送分离数据（ASB 状态等）直到本函数执行结束。所以，当从打印机接收执行结果时推荐使用本函数。

> NOTE
> 考虑到扩展性和多功能性，推荐使用 BiDirectIOEx 而不是 BiDirectIO 函数。
> 有关 ESC/POS 命令请联系经销商。

语法

nErr = **BiDirectIOEx** (*nHandle, writeLen, writeCmd, readLen, readBuff, Timeout, nullTerminate, option*)

请参照下一变量。

变量

nHandle: 指定句柄。此为 INT 类型。

writeLen: 指定写入打印机的数据长度。当为 “0”时则不写入打印机。此为 DWORD 类型。

writeCmd: 指定写入打印机的数据 （ESC/POS 命令）。此为 LPBYTE 类型。

readLen: 指定从打印机读取的数据长度。
当打印机需要命令执行结果时指定。
当不需要时指定为 “0”。此为 LPDWORD 类型。

readBuff: 指定保存从打印机读取的数据的缓冲。
此为 LPBYTE 类型。

Timeout: 以 ms （毫秒）为单位指定超时时间。此为 DWORD 类型。

nullTerminate:在 “True”的情况下，当从打印机接收 NULL 时读取结束。此时，指定 readBuff 大小至 readLen。
在 “FALSE”的情况下，在 readLen 中指定的数据长度被读取，或发生超时错误前数据从打印机被读取。

option: 控制 ASB 禁止命令。此为 BYTE 类型。

值	说明
0	写入数据前发送 ASB 禁止命令，读取数据后启用 ASB。
1	不发送 ASB 禁止命令或 ASB 启用命令。

> NOTE
> 请确保 readBuff 的大小与 readLen 中指定的长度一致或大于其值。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_NO_MEMORY	-50	没有足够的内存空间
ERR_HANDLE	-60	指定的句柄无效
ERR_TIMEOUT	-70	超时错误
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_PARAM	-90	参数错误
ERR_BUFFER_OVER_FLOW	-140	缓冲区空间不足
ERR_EXEC_FUNCTION	-310	由于 Status API 正被其它应用程序使用， 所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_RESET	-400	打印机正在重启，无法调用
ERR_LOCKED	-1000	打印机被锁定。

NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

通过确认 *nErr* 的返回值来确认函数是否正确执行，或通过确认打印机的运行状况来确认命令是否正确执行。如果从打印机获取了执行结果 （指定 *readLen*），确认该执行结果。
调用此函数时，根据打印机的状态执行如下表所示的操作。

打印机状态	操作
联机	返回 “SUCCESS”至 *nErr*。 执行命令。
脱机	当在 *Timeout* 指定的时间内成功结束与打印机的通讯时，会返回“SUCCESS”。 当在 *Timeout* 指定的时间内与打印机通讯失败时，会返回“ERR_TIMEOUT”。
电缆移除 / 电源关闭	返回 “ERR_ACCESS”至 *nErr*。
打印中	返回 “ERR_LOCKED”至 *nErr*。

注意事项

- 虽然指定给读 / 写操作的最大数据长度为 2GB，指定需要的最小数据长度。
- 在监控打印机状态时，请不要使用本函数发送 ASB 状态传输的无效命令。可获取并发状态。
- ASB （自动状态通知）禁止命令确保在发送要求打印机回应的命令时不会获取无关数据。如果您不使用 ASB 禁止命令，请确保程序考虑了对于无关数据的接收。
- 指定接收缓冲使用本函数处理从打印机接收的数据，或使用同一进程作为监测载体（BiGetStatus 函数等）处理数据。请参照下表。

传输命令	接收指定缓冲	接收缓冲	监测载体的操作
获取状态命令	是	保存 ASB 状态来接收缓冲	不回调 不更新 状态
	否	-	不回调 不更新 状态
带有其它打印机回应的命令	是	在接收缓冲中输入打印机回应	不影响监测载体
	否	-	可能产生不正常的回调
没有其它打印机回应的命令	是	发生超时错误	不影响监测载体
	否	-	不影响监测载体

BiResetPrinter

重启状态监测打印机。

NOTE

- 在打印中取消调用的打印任务。
- 连接串行接口的 TM 打印机无法重置。

语法

nErr = **BiResetPrinter** (*nHandle*)

通过 *nHandle* 指定句柄。

变量

nHandle: 指定句柄。此为 INT 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_NOT_SUPPORT	-100	不支持
ERR_EXEC_FUNCTION	-310	另一 Status API 正被使用，所以无法调用
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_RESET	-400	打印机正在重启，无法调用
ERR_LOCKED	-1000	打印机被锁定。

NOTE

有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

通过确认 *nErr* 的返回值来确认本函数执行的正确性，或重置打印机确认打印机处于联机状态（确认 ASB 状态）。

NOTE

本函数执行后， 15 秒内打印机无法接收打印命令。如果在这段期间执行打印，打印任务被发送至后台，打印操作将在上述时间后被执行。

调用此函数时，根据打印机的状态执行如下表所示的操作。

打印机状态	操作
联机	返回 “SUCCESS”至 *nErr* 并重置。
脱机	返回 “SUCCESS”至 *nErr* 并重置。
电缆移除 / 电源关闭	返回 “ASB_NO_RESPONSE”至 ASB 状态，不重置打印机。
打印中	取消打印任务并重启打印机。

BiForceResetPrinter

强制重置其状态正在被监测的 TM 打印机。
在多线程 / 进程 / 用户的环境下，也可进行 TM 打印机的复位。其他程序以 BiLockPrinter 独享访问时，也可进行 TM 打印机的复位。网络打印机的连接断开时，重新连上后，需要等待一定的时间才可以执行印刷。使用该 API 则可以缩短该时间。

CAUTION

- 即使其他的线程 / 进程 / 用户正在印刷中，也可强制性进行复位，所以请慎用该 API。
- 即使打印机正在印刷中，此 API 也可强制打印机重置，打印数据将被删除。
- 连接串行接口的 TM 打印机无法重置。

语法

nErr = **BiForceResetPrinter** (*nHandle*)
通过 *nHandle* 指定句柄。

变量

nHandle: 指定句柄。此为 INT 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_NOT_SUPPORT	-100	不支持
ERR_EXEC_FUNCTION	-310	另一 Status API 正被使用，所以无法调用
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。

NOTE

有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

BiCancelError

如果打印机发生可恢复的错误，则在清除错误原因后执行本函数，且 TM 打印机可从错误中恢复。

> NOTE 如果打印机在传输数据时发生可恢复的错误，则调用此函数不能恢复打印机状态。在此情况下，清除错误原因后使用 BiResetPrinter 并从错误中恢复打印机状态。

语法

nErr = **BiCancelError** (*nHandle*)

通过 *nHandle* 指定句柄。

变量

nHandle: 指定句柄。此为 INT 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_NOT_SUPPORT	-100	不支持
ERR_EXEC_FUNCTION	-310	由于 Status API 正被其它应用程序使用， 所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_RESET	-400	打印机正在重启，无法调用
ERR_LOCKED	-1000	打印机被锁定。

> NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

如下表所示使用 BiCancelError。

错误	对应方案
盖板打开错误	关闭盖板后调用本函数。
自动裁纸器错误	清除裁纸器周围的打印纸并关闭盖板后调用本函数。

BiGetType

获取打印机的类型 ID。

> NOTE 有关可获取的类型 ID 的信息，请联系经销商。

语法

nErr = **BiGetType** (*nHandle*, *typeID*, *font*, *exrom*, *special*)

如果您指定一个句柄至 nHandle，一个类型 ID 被设置到 typeID，设备字体被设置到 font。打印机的特殊 ID 返回至 special。

变量

nHandle: 指定句柄。此为 INT 类型。
typeID: 打印机的一个类型 ID 将被设置。此为 LPBYTE 类型。
font: 设备字体将被设置。此为 LPBYTE 类型。
exrom: 此项不可用。此为 LPBYTE 类型。
special: 打印机的一个特殊 ID 将被设置。此为 LPBYTE 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_TIMEOUT	-70	超时错误
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_PARAM	-90	参数错误
ERR_NOT_SUPPORT	-100	不支持
ERR_EXEC_FUNCTION	-310	Status API 正被其它应用程序使用，所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_RESET	-400	打印机正在重启，无法调用
ERR_LOCKED	-1000	打印机被锁定。

> NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

BiGetStatus

获取当前打印机状态 （ASB 状态）。

语法

nErr = **BiGetStatus** (*nHandle* , *lpStatus*)

通过 *nHandle* 指定句柄。返回 ASB 状态至 *lpStatus*。

变量

nHandle: 指定句柄。此为 INT 类型。

lpStatus: 返回保存至 Status API 的 ASB 状态。此为 LPDWORD 类型。
ASB 状态由 4 个字节构成。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_PARAM	-90	参数错误
ERR_EXEC_FUNCTION	-310	另一 Status API 正被使用，所以无法调用

NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

有关 TM-T81 可获取的 ASB 状态，请参照第 87 页 “机型信息”。

BiGetRealStatus

获取当前打印机状态 （ASB 状态）。

语法

nErr = **BiGetRealStatus** (*nHandle* , *lpStatus*)

变量

nHandle: 指定句柄。此为 INT 类型。

lpStatus: 返回保存至 Status API 的 ASB 状态。此为 LPDWORD 类型。
ASB 状态由 4 个字节构成。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_PARAM	-90	参数错误
ERR_EXEC_FUNCTION	-310	另一 Status API 正被使用，所以无法调用
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_RESET	-400	打印机正在重启，无法调用
ERR_LOCKED	-1000	打印机被锁定。

NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

本函数发送获取 ASB 状态的命令至打印机，并接收调用函数后获取的状态。那就是为什么即使打印已经结束还未获取 ASB_PRINTSUCCESS 的原因。同时，当打印机电源关闭时，由于返回了 ERR_ACCESS， ASB_NO_RESPONSE 也未能获取。

有关 TM-T81 可获取的 ASB 状态，请参照第 87 页 “机型信息”。

BiSetStatusBackFunction

当打印机状态改变时，使用回调函数自动获取打印机状态 （ASB 状态）。

NOTE 当开发环境是 VB 时此项不可用。

语法

nErr = **BiSetStatusBackFunction** (*nHandle*, *int (CALLBACK EXPORT *pStatusCB) (DWORD dwStatus)*)

变量

nHandle: 指定句柄。此为 INT 类型。

int (CALLBACK EXPORT *pStatusCB)(DWORD dwStatus):
指定回调函数的定义地址。

dwStatus: 返回保存至 Status API 的 ASB 状态。此为 DWORD 类型。
ASB 状态由 4 个字节构成。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_PARAM	-90	参数错误
ERR_EXIST	-210	指定的数据已经存在。
ERR_EXEC_FUNCTION	-310	另一 Status API 正被使用，所以无法调用

NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

调用此函数设置打印机状态至 *dwStatus* 并调用回调函数。当打印机状态改变时，新信息会被自动设置到 *dwStatus* 并调用回调函数。使用 BiCancelStatusBack 取消该函数。
有关 TM-T81 可获取的 ASB 状态，请参照第 87 页 “机型信息”。

NOTE 在注册的回调函数内不可使用 Status API。

BiSetStatusBackFunctionEx

当打印机状态改变时，使用回调函数自动获取打印机状态 （ASB 状态）。
指定启动回调的打印机端口，包括函数 BiSetStatusBackFunction。

NOTE 当开发环境是 VB 时此项不可用。

语法

nErr = **BiSetStatusBackFunctionEx** (*nHandle*, *int (CALLBACK EXPORT *pStatusCB)*
(DWORD dwStatus, LPSTR lpcPortName)

变量

nHandle: 指定句柄。此为 INT 类型。
int (CALLBACK EXPORT *pStatusCB)(DWORD dwStatus, LPSTR lpcPortName):
指定回调函数的定义地址。
dwStatus: 返回保存至 Status API 的 ASB 状态。ASB 状态由 4 个字节构成。此为 DWORD 类型。
lpcPortName: 返回启动回调的打印机端口名称。此为 LPSTR 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_PARAM	-90	参数错误
ERR_EXIST	-210	指定的数据已经存在。
ERR_EXEC_FUNCTION	-310	另一 Status API 正被使用，所以无法调用

NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

调用此函数设置打印机状态至 *dwStatus* 并调用回调函数。当打印机状态改变时，新信息会被自动设置到 *dwStatus* 并调用回调函数。使用 BiCancelStatusBack 取消该函数。
有关 TM-T81 可获取的 ASB 状态，请参照第 87 页 “机型信息”。

NOTE 在注册的回调函数内不可使用 Status API。

BiSetStatusBackWnd

当打印机状态改变时，自动产生一个点击事件并获取打印机状态 （ASB 状态）。

语法

nErr = **BiSetStatusBackWnd** (*nHandle*, *hWnd*, *lpStatus*)

通过 *nHandle* 指定句柄。返回 ASB 状态至 *lpStatus*。

变量

nHandle: 指定句柄。此为 INT 类型。

hWnd: 指定产生点击事件的按钮的 Window 句柄。
此为 Long 类型。

lpStatus: 返回保存至 Status API 的 ASB 状态。此为 LPDWORD 类型。
ASB 状态由 4 个字节构成。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_PARAM	-90	参数错误
ERR_EXIST	-210	指定的数据已经存在。
ERR_EXEC_FUNCTION	-310	另一 Status API 正被使用，所以无法调用

NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

有关 TM-T81 可获取的 ASB 状态，请参照第 87 页 “机型信息”。

NOTE 指定的 Window 句柄不可使用 Status API。

BiCancelStatusBack

使用 BiSetStatusBackFunction， BiSetStatusBackFunctionEx 或 BiSetStatusBackWnd 函数退出调用的自动状态通知请求进程。

语法

nErr = **BiCancelStatusBack** (*nHandle*)

通过 *nHandle* 指定句柄。

变量

nHandle: 指定句柄。此为 INT 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_EXEC_FUNCTION	-310	另一 Status API 正被使用，所以无法调用

NOTE 即使在自动状态通知请求进程未被注册时执行，也会返回 “SUCCESS”。有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

BiPowerOff

更新维护计数器并准备关闭打印机电源。
不可关闭打印机电源。

> NOTE 处于联机恢复待机状态时无法调用。

语法

nErr = **BiPowerOff** (*nHandle*)

通过 *nHandle* 指定句柄。

变量

nHandle: 指定句柄。此为 INT 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_NO_MEMORY	-50	没有足够的内存空间
ERR_HANDLE	-60	指定的句柄无效
ERR_TIMEOUT	-70	超时错误
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_NOT_SUPPORT	-100	不支持
ERR_EXEC_FUNCTION	-310	另一 Status API 正被使用，所以无法调用
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_LOCKED	-1000	打印机被锁定。

> NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

TM 打印机执行如下步骤。

- 存储维护计数器值。
- 将接口置为 BUSY 状态。
- 将 TM 打印机置于关闭待机状态。

BiGetCounter

获取维护计数器值。

NOTE

- 有关可获取的计数器编号和维护计数器的信息，请联系经销商。
- 根据打印机不同，维护计数器可能不可用。在这种情况下，发生超时错误。
- 确认在调用本函数前 ASB 状态处于联机状态。

语法

nErr = **BiGetCounter** (*nHandle, readno*, *readcounter*)

通过 *nHandle* 指定句柄。 *readno* 指定获取的维护计数器编号，且维护计数器值返回至 *readcounter*。

变量

nHandle: 指定句柄。此为 INT 类型。

readno: 指定获取的维护计数器编号。此为 WORD 类型。

readcounter: 返回维护计数器。此为 LPDWORD 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_TIMEOUT	-70	超时错误
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_PARAM	-90	参数错误
ERR_NOT_SUPPORT	-100	不支持
ERR_EXEC_FUNCTION	-310	Status API 正被其它应用程序使用，所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_RESET	-400	打印机正在重启，无法调用
ERR_LOCKED	-1000	打印机被锁定。

NOTE

有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

有两种维护计数器；一种是可以由用户重置，另一种是无法重置的结合计数器。
调用此函数时，根据打印机的状态执行如下表所示的操作。

打印机状态	操作
联机	返回 “SUCCESS”至 *nErr*。获取维护计数器值。
脱机	返回 “ERR_TIMEOUT”至 *nErr*。不获取维护计数器值。
电缆移除 / 电源关闭	返回 “ERR_ACCESS”至 *nErr*。 不获取维护计数器值。
打印中	返回 “ERR_ACCESS”至 *nErr*。 不获取维护计数器值。

BiResetCounter

重置维护计数器。

NOTE

- 有关可获取的计数器编号和维护计数器的信息，请联系经销商。
- 根据打印机不同，维护计数器可能不可用。在这种情况下，发生超时错误。
- 确认在调用本函数前 ASB 状态处于联机状态。

语法

nErr = **BiResetCounter** (*nHandle, readno*)

通过 *nHandle* 指定句柄。指定重置到 *readno* 的维护计数器编号。

变量

nHandle: 指定句柄。此为 INT 类型。

readno: 指定要重置的维护计数器编号。此为 WORD 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_TIMEOUT	-70	超时错误
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_PARAM	-90	参数错误
ERR_NOT_SUPPORT	-100	不支持
ERR_EXEC_FUNCTION	-310	Status API 正被其它应用程序使用，所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_RESET	-400	打印机正在重启，无法调用
ERR_LOCKED	-1000	打印机被锁定。

NOTE

有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

如果 *nErr* 的返回值和重置维护计数器后 BiGetCounter 获取的值均是 “SUCCESS”，则可确认此次执行正常。

调用此函数时，根据打印机的状态执行如下表所示的操作。

打印机状态	操作
联机	返回 “SUCCESS”至 *nErr*。重置维护计数器。
脱机	返回 “ERR_TIMEOUT”至 *nErr*。不重置维护计数器。
电缆移除 / 电源关闭	返回 “ERR_ACCESS”至 *nErr*。不重置维护计数器。
打印中	返回 “ERR_ACCESS”至 *nErr*。不重置维护计数器。

BiGetPrnCapability

通过打印机 ID 获取指定的打印机信息。

> NOTE 有关可获取的打印机容量的信息，请联系经销商。

语法

nErr = **BiGetPrnCapability** (*nHandle*, *prnID*, *pBuffSize*, *pBuff*)

通过 *nHandle* 指定句柄，并指定待获取打印机信息至 *prnID*。通过 *pBuffSize* 指定内存大小来设置打印机信息，并通过 *pBuff* 指定内存地址来设置打印机信息。

变量

nHandle: 指定句柄。此为 INT 类型。

prnID: 指定待获取打印机信息。此为 BYTE 类型。

pBuffSize: 指定内存大小来设置打印机信息 （1 至 80）。在调用本函数后返回实际读取的数据大小。在缓冲区容量不足的情况下，返回需要的字节大小。此为 LPBYTE 类型。

pBuff: 指定内存地址来设置打印机信息。此为 LPBYTE 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_TIMEOUT	-70	超时错误
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_PARAM	-90	参数错误
ERR_BUFFER_OVER_FLOW	-140	缓冲区空间不足
ERR_EXEC_FUNCTION	-310	Status API 正被其它应用程序使用，所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_RESET	-400	打印机正在重启，无法调用
ERR_LOCKED	-1000	打印机被锁定。

> NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

BiOpenDrawer

打开货币纸盒。

NOTE 即使当打印机脱机时，打开货币纸盒。

语法

nErr = **BiOpenDrawer** (*nHandle*, *drawer, pulse*)

通过 *nHandle* 指定句柄。在 *drawer* 中指定要打开的货币纸盒，并指定时间直至在 *pulse* 中货币纸盒打开。

变量

nHandle: 指定句柄。此为 INT 类型。

drawer: 指定要打开的货币纸盒。此为 BYTE 类型。

宏定义 （常量）	值	说明
EPS_BI_DRAWER_1	1	打开 1 号货币纸盒
EPS_BI_DRAWER_2	2	打开 2 号货币纸盒

pulse: 指定时间直至货币纸盒打开。此为 BYTE 类型。

宏定义 （常量）	值	说明
EPS_BI_PLUSE_100	1	100 毫秒后运行货币纸盒
EPS_BI_PLUSE_200	2	200 毫秒后运行货币纸盒
EPS_BI_PLUSE_300	3	300 毫秒后运行货币纸盒
EPS_BI_PLUSE_400	4	400 毫秒后运行货币纸盒
EPS_BI_PLUSE_500	5	500 毫秒后运行货币纸盒
EPS_BI_PLUSE_600	6	600 毫秒后运行货币纸盒
EPS_BI_PLUSE_700	7	700 毫秒后运行货币纸盒
EPS_BI_PLUSE_800	8	800 毫秒后运行货币纸盒

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_PARAM	-90	参数错误
ERR_NOT_SUPPORT	-100	不支持
ERR_EXEC_FUNCTION	-310	Status API 正被其它应用程序使用，所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_RESET	-400	打印机正在重启，无法调用
ERR_LOCKED	-1000	打印机被锁定。

NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

调用此函数时，根据打印机的状态执行如下表所示的操作。

打印机状态	操作
联机	返回 “SUCCESS”至 *nErr*。打开货币纸盒。
脱机	返回 “SUCCESS”至 *nErr*。打开货币纸盒。
电缆移除 / 电源关闭	返回 “ERR_ACCESS”至 *nErr*。不打开货币纸盒。

BiSendDataFile

指定命令定义文件来定义传输命令 （ESC/POS 命令）。

> NOTE
> 以指定的格式指定命令定义文件。
> 有关 ESC/POS 命令请联系经销商。

语法

nErr = **BiSendDataFile** (*nHandle, lpcFileName*)

通过 *nHandle* 指定句柄，并通过 *lpcFileName* 指定命令定义文件。

变量

nHandle: 指定句柄。此为 INT 类型。

lpcFileName: 指定命令定义文件名。此为 LPCSTR 类型。

返回值

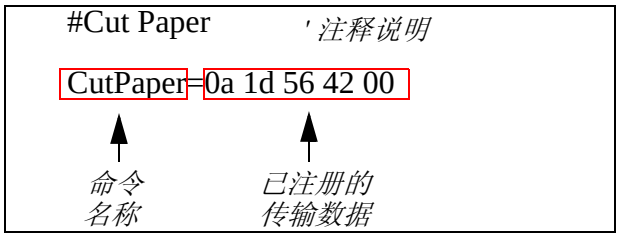
返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_NO_MEMORY	-50	没有足够的内存空间
ERR_HANDLE	-60	指定的句柄无效
ERR_PARAM	-90	参数错误
ERR_EXEC_FUNCTION	-310	另一 Status API 正被使用，所以无法调用

> NOTE
> 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

使用以下格式描述命令定义文件。



注意事项

- 在符号 “#”后运用字符串作为注释说明。
- 符号 “=”左边的字符串是写入打印机的 “命令名称”，符号 “=”右边的字符串是 “已注册的传输数据”。
- 确保使用括号符 （" "）表示字符串。
- 二进制数据表示成两位的十六进制。
- “命令名称”的最多字节是 33 个字节 （ANK 字体时是 33 个字符）。
- “已注册的传输数据”的最多字节是 10， 240 个字节。然而，“已注册的传输数据”的字节不是 “ASCIIZ 字符串”的字节而是转换成二进制数据后的字节。请参照以下示例。
  例如)　ABC="ABC"　：“已注册的传输数据”的最多字节是 3 个字节。
  ABC="ABC" 0D 0A ：“已注册的传输数据”的最多字节是 5 个字节。
  ABC=41 42 43 0D 0A：“已注册的传输数据”的最多字节是 5 个字节。
- 如果命令名称若已经注册，则停止命令注册进程并返回一个错误。
- 可被注册的命令的数量由系统可用的内存大小决定。
- 调用 BiCloseMonPrinter 函数退出已注册的命令数据。

BiDirectSendRead

执行由 BiSendDataFile 定义的命令 （ESC/POS 命令）。

语法

nErr = **BiDirectSendRead** (*nHandle, lpcCmdName, lpcReadName, readLen, pReadBuf, Timeout, nullTerminate*)

请参照下一变量。

变量

nHandle:	指定句柄。此为 INT 类型。
lpcCmdName:	指定命令定义文件名称的 “命令名称”。此为 LPCSTR 类型。
lpcReadName:	指定从打印机读取的 “接收数据类型名称”。 请参照第 60 页 “BiSendDataFile”。此为 LPCSTR 类型。
readLen:	指定从打印机读取的数据长度。当为 “0”时则不写入打印机。读取数据时返回数据长度。此为 LPDWORD 类型。
preadBuff:	指定保存从打印机读取的数据的缓冲区。 此为 LPBYTE 类型。
Timeout:	以 ms （毫秒）为单位指定超时时间。此为 DWORD 类型。
nullTerminate:	在 “TRUE”的情况下，当从打印机接收到 NULL 时数据读取结束。此时，指定 readBuff 大小至 readLen。 在 “FALSE”的情况下，在 readLen 中指定的数据长度被读取，或发生超时错误前数据从打印机被读取。

NOTE

确保 preadBuff 的大小大于或等于 readLen 中指定的长度。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_NO_MEMORY	-50	没有足够的内存空间
ERR_HANDLE	-60	指定的句柄无效
ERR_TIMEOUT	-70	超时错误
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_PARAM	-90	参数错误
ERR_EXEC_FUNCTION	-310	Status API 正被其它应用程序使用，所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_RESET	-400	打印机正在重启，无法调用
ERR_LOCKED	-1000	打印机被锁定。

NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

本函数可以预先指定在命令定义文件中已指定的名称 （宏名称）。打印机的回应可以指定为下表所示的数据类型。

数据类型	说明
ASB	自动状态传输
ASB Extended	扩展状态的自动状态传输
Ptr Info Byte	打印机 ID 信息
Ptr Info String	打印机信息 B
Power OFF	电源关闭通知
Power ON	电源开启通知
Realtime	实时状态传输
Buffer Clear	清空缓冲区
Slip Remaining	单纹页剩余打印区的点数传输
NVM Image Size	NV 字符区总容量的传输
NVM Image Free	NV 字符区剩余容量的传输
NVM Image Keys	已定义的 NV 字符的关键编码列表的传输
NVM Image List	传输保存至 NV 内存的图像读取结果的数据 ID 列表
NVM User Used	已使用容量的传输 （使用区域的字节数）
NVM User Free	剩余容量的传输 （未使用区域的字节数）
NVM User Get	传输指定记录的存储数据
NVM User Keys	传输存储数据的关键编码列表

数据类型	说明
NVM Set Mode	传输用户设置模式的传输通知
NVM Get Mswitch	内存开关值的传输
NVM Set Size	定制值的传输
Ptr Info Type A	打印机信息 A
Test Print	执行测试打印
RAM Image Free	传输下载图像区域的剩余区域
RAM Image Keys	传输已定义的下载图像的关键编码列表
OfflineCode Bit	脱机回应的传输 （位格式）
OfflineCode Data	脱机回应的传输 （数据格式）
ProcessID	传输进程 ID 回应
Data Types	说明
Buffer Clear24	缓冲区清空 24
Other	上述数据不可用

NOTE

有关 ESC/POS 命令请联系经销商。

BiSetDefaultEchoTime

配置网络打印机的回应确认频率和初始化发生单次超时的时间值。

NOTE 只能用于通过 Ethernet 连接的情况。

语法

nErr = **BiSetDefaultEchoTime** (*Count*, *Timeout*)

配置打印机回应确认频率至 *Count* 和发生单次超时的时间至 *Timeout*。

变量

Count: 配置回应确认频率 （1 至 255）。此为 BYTE 类型。

Timeout: 以 ms （毫秒）为单位配置发生单次超时的时间 （1 至 65535）。此为 WORD 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_PARAM	-90	参数错误
ERR_EXEC_FUNCTION	-310	Status API 正被其它应用程序使用，所以此函数不可用。
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。

NOTE
- 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。
- 当共享打印机从一台无 PortHandler 的客户端被访问时，打印机会返回 ERR_PH_NOT_EXIST。

注释说明

安装 Status API 后，回应确认频率立即被设置为三次，超时时间也设置为一秒。重启计算机后，通过 API 的配置将有效。通过 API 的配置将对所有连接至计算机的 TM 打印机有效 （w/ Ethernet 端口）。

BiSetEtherEchoTime

在 Status API 可用后，配置一次网络打印机的回应确认频率值和超时时间。

> NOTE 只能用于通过 Ethernet 连接的情况。

语法

nErr = **BiSetEtherEchoTime** (*nHandle* , *Count*, *Timeout*)

通过 *nHandle* 指定句柄。配置打印机回应确认频率至 *Count* 和发生单次超时的时间至 *Timeout*。

变量

nHandle: 指定句柄。此为 INT 类型。

Count: 配置回应确认频率 （1 至 255）。此为 BYTE 类型。

Timeout: 以 ms （毫秒）为单位配置发生单次超时的时间 （1 至 65535）。此为 WORD 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功
ERR_HANDLE	-60	指定的句柄无效
ERR_ACCESS	-80	打印机无法执行读 / 写操作
ERR_PARAM	-90	参数错误
ERR_NOT_SUPPORT	-100	不支持
ERR_EXEC_FUNCTION	-310	另一 Status API 正被使用，所以无法调用
ERR_PH_NOT_EXIST	-340	PortHandler 不运行，或 PortHandler 客户端和服务器之间的通讯错误。
ERR_LOCKED	-1000	打印机被锁定。

> NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

注释说明

如果本函数未设置某个值，则使用 BiSetDefaultEchoTime 函数中设置的值。

BiSetReadWaitTimeOut

这是一个兼容的 API。本 API 本身无此功能。

语法

nErr = **BiSetReadWaitTimeOut** (*nHandle* , *wTimeOut*)

变量

nHandle: 指定句柄。此为 INT 类型。
wTimeOut: 未使用。此为 WORD 类型。

返回值

返回下表所示的 Status API 执行错误 （值）至 INT 类型定义的变量 （nErr）。当此函数被成功调用时，会返回 “SUCCESS”（宏定义）。

宏定义 （常量）	值	说明
SUCCESS	0	成功

NOTE 有关 Status API 执行错误的恢复信息，请参照第 22 页 “Status API 错误及对应方案”。

.NET 参考

本章解释说明了 .Net 环境下使用的 Status API 和语法。

属性

IsValid

获取打印机的打开状态。
访问： 只读
数据类型：System.Boolean

解释说明

返回以下两种值之一。
true: 成功打开。
false: 未打开或打开失败。

LastError

获取最新执行的 API 的错误编码。
访问： 只读
数据类型：com.epson.pos.driver.ErrorCode

解释说明

由于此模块保留最新执行的 API，所以可在任何时候获取错误编码。
由于提供给属性的 API 无法返回错误编码，所以此方法用于判断执行的成功或失败。
对应错误的错误编码可能发生在所有的 API 中。有关详情，请参见第 23 页 “Status API 执行错误”。

Status

调用 APD4StatusAPI 中的 BiGetSatus 并获取打印机的当前状态。
访问： 只读
数据类型：com.epson.pos.driver.ASB

解释说明

com.epson.pos.driver.ASB 中定义的常量可用于获得此值。有关详情，请参见第 46 页 “BiGetStatus”。

方法

OpenMonPrinter

开始控制指定的打印机。
调用 APD4 Status API 中的 BiOpenMonPrinter。有关详情，请参见第 28 页 “BiOpenMonPrinter”。

原型

ErrorCode ***OpenMonPrinter*** (OpenType type, String name)

参数

OpenType type: 指定名称的名称类型。com.epson.pos.driver.OpenType 中定义的常量可用于获得此值。
String name: 开始控制指定的打印机。

返回值不同于 APD4 Status API 的返回编码

本方法只返回一个在 com.epson.pos.driver.ErrorCode 中定义的错误编码。（不返回句柄。）

CloseMonPrinter

停止控制指定的打印机。
调用 APD4 Status API 中的 BiCloseMonPrinter。有关详情，请参见第 30 页 “BiCloseMonPrinter”。

原型

ErrorCode ***CloseMonPrinter*** ( )

LockPrinter

占用打印机。
调用 APD4 Status API 中的 BiLockPrinter。有关详情，请参见第 31 页 “BiLockPrinter”。

原型

ErrorCode ***LockPrinter***(int timeout)

参数

int timeout: 超时时间 （以毫秒为单位）。

UnlockPrinter

停止占用打印机。
调用 APD4 Status API 中的 BiUnLockPrinter。有关详情，请参见第 33 页“BiUnlockPrinter”。

原型

ErrorCode ***UnlockPrinter*** ( )

SetMonInterval

指定 Status API 的时间间隔来读取打印机的状态。
调用 APD4 Status API 中的 BiSetMonInterval。有关详情，请参见第 34 页“BiSetMonInterval”。

原型

ErrorCode ***SetMonInterval***(int noPrnInterval, int prnInterval)

参数

int noPrnInterval: 未使用
int prnInterval: 打印机监测时间间隔

SetMonEtherInterval

指定 Status API 的时间间隔来读取打印机的状态。
调用 APD4 Status API 中的 BiSetMonEtherInterval。有关详情，请参见第 35 页 “BiSetMonEtherInterval”。

原型

ErrorCode ***SetMonEtherInterval***(int EtherInterval)

参数

int EtherInterval: 网络打印机监测时间间隔

DirectIOEx

发送指定的数据至打印机后，从打印机接收指定长度的数据。
调用 APD4 Status API 中的 BiDirectIOEx。有关详情，请参见第 38 页 “BiDirectIOEx”。

原型

1. ErrorCode ***DirectIOEx***(byte[] writeCmd, ref byte[] readBuff, int timeout, bool nullTerminate, byte option)
   描述 : 发送 ESC/POS 命令至 TM 打印机，并从打印机接收执行结果 （二进制数据）。
2. ErrorCode ***DirectIOEx***(byte[] writeCmd, out String response, int timeout, byte option)
   描述 : 发送 ESC/POS 命令至 TM 打印机，并从打印机接收执行结果 （字符串数据）。
3. ErrorCode ***DirectIOEx***(byte[] writeCmd, int timeout)
   描述 : 仅发送 ESC/POS 命令至 TM 打印机。从打印机既没接收到执行结果，也没接收到 ASB 状态。

参数

byte[] writeCmd: 要发送至打印机的数据
ref byte[] readBuff: 从打印机接收到的数据
int timeout: 数据传输和接收的超时时间 （以毫秒为单位）
bool nullTerminate: 当接收到 NULL 时判断是否结束接收
byte option: 在 “True”的情况下，当从打印机接收到 NULL 时读取结束。此时，指定 readBuff 大小至 readLen。
在 “FALSE”的情况下，在 readLen 中指定的数据长度被读取，或发生超时错误前数据从打印机被读取。
out String response: 从打印机接收的数据 （要转换成字符串）

ResetPrinter

重置打印机。当在打印过程中重置打印机时，取消打印任务并执行打印机重置。
调用 APD4 Status API 中的 BiResetPrinter。有关详情，请参见第 41 页 “BiResetPrinter”。

原型

ErrorCode ***ResetPrinter*** ( )

ForceResetPrinter

强制重置其状态正在被监测的 TM 打印机。也可以重置被 LockPrinter 占用的 TM 打印机。这样也重置正在打印的 TM 打印机。使用此 API 时请小心。
调用 APD4 Status API 中的 BiForceResetPrinter。有关详情，请参见第 43 页 “BiForceResetPrinter”。

原型

ErrorCode ***ForceResetPrinter*** ( )

CancelError

调用 APD4 Status API 中的 BiCancelError。从打印机的可恢复错误中恢复。
有关详情，请参见第 44 页 “BiCancelError”。

原型

ErrorCode ***CancelError*** ( )

GetType

获取打印机的类型 ID。对于某些机型一些信息可能不能获取。此情况下设置为。
调用 APD4 Status API 中的 BiGetType。有关详情，请参见第 45 页 “BiGetType”。

原型

ErrorCode ***GetType***(out byte typeid, out byte font, out byte exrom, out byte euspecial)

参数

out byte typeid: 打印机的类型 ID。
out byte font: 安装于打印机内的字体
out byte exrom: 打印机的扩展闪存的容量。
out byte euspecial: 打印机的指定 ID

GetRealStatus

获取打印机的最新状态。应使用对应于 ASB 状态内容的状态中的个别位和定义于 com.epson.pos.driver. ASB 中的常量。
调用 APD4 Status API 中的 BiGetRealStatus。有关详情，请参见第 47 页 “BiGetRealStatus”。

原型

ErrorCode ***GetRealStatus***(out ASB asb)

参数

out ASB asb: 打印机的当前状态

SetStatusBack

通过 StatusCallback/StatusCallbackEx 事件开始状态通知。
调用 APD4 Status API 中的 BiSetStatusBackFunctionEx。有关详情，请参见第 49 页 “BiSetStatusBackFunctionEx”。

原型

ErrorCode ***SetStatusBack*** ( )

CancelStatusBack

通过 StatusCallback/StatusCallbackEx 事件停止状态通知。
调用 APD4 Status API 中的 BiCancelStatusBack。有关详情，请参见第 51 页 “BiCancelStatusBack”。

原型

ErrorCode ***CancelStatusBack*** ( )
ErrorCode ***CancelStatusBack*** ( )

PowerOff

执行打印机的电源关闭进程。
调用 APD4 Status API 中的 BiPowerOff。有关详情，请参见第 52 页 “BiPowerOff”。

原型

ErrorCode ***PowerOff*** ( )

GetCounter

读取维护计数器。
调用 APD4 Status API 中的 BiGetCounter。有关详情，请参见第 53 页 “BiGetCounter”。

原型

1. ErrorCode ***GetCounter***(CounterIndex counter, bool cumulative, out int value)
   描述 : 通过 CounterIndex counter 和 bool cumulative 的组合，计算计数器编号并获取计数器的值。
2. ErrorCode ***GetCounter***(byte counter, out int value)
   描述 : 获取由 byte counter 指定的计数器的值。

参数

CounterIndex counter:	com.epson.pos.driver.CounterIndex 中定义的维护计数器 numberConstants 应该用于获取此值。
bool cumulative:	维护计数器数是否由累积计数器指定。 true: 累积计数器 false: 重置计数器
out int value:	维护计数器值
byte counter:	维护计数器编号

ResetCounter

重置维护计数器。
调用 APD4 Status API 中的 BiResetCounter。有关详情，请参见第 55 页 “BiResetCounter”。

原型

当指定由 com.epson.pos.driver.CounterIndex 定义的作为常量的一个变量时，使用下列 “1” 语法。当指定其它的变量时，使用 “2” 语法。

1. ErrorCode ***ResetCounter***(CounterIndex counter)
2. ErrorCode ***ResetCounter***(byte counter)

参数

CounterIndex counter:	com.epson.pos.driver.CounterIndex 中定义的维护计数器 numberConstants 应该用于获取此值。
byte counter:	维护计数器编号

GetPrnCapability

获取由打印机 ID 指定的打印机的信息。
调用 APD4 Status API 中的 BiGetPrnCapability。有关详情，请参见第 57 页 “BiGetPrnCapability”。

原型

1. ErrorCode ***GetPrnCapability***(byte printerID, out byte[] data)
   描述 : 获取由打印机 ID 指定的 TM 打印机的信息 （二进制数据）。
2. ErrorCode ***GetPrnCapability***(byte printerID, out String data)
   描述 : 获取由打印机 ID 指定的 TM 打印机的信息 （字符串数据）。

参数

byte printerID: 获取信息中的打印机的 ID。
out byte[] data: 打印机信息
out String data: 打印机信息

OpenDrawer

激活货币纸盒。打印机处于脱机状态时也可使用。
调用 APD4 Status API 中的 BiOpenDrawer。有关详情，请参见第 58 页 “BiOpenDrawer”。

原型

ErrorCode ***OpenDrawer***(Drawer drawer, Pulse pulse)

参数

Drawer drawer: com.epson.pos.driver.Drawer 中定义的 openedConstants 的货币纸盒应该用于获取此值。
Pulse pulse: com.epson.pos.driver 中定义的 drawerConstants 的激活时间间隔上升。Pulse 应该用于获取此值。

SendDataFile

使用命令定义文件注册命令。有关命令定义文件的格式，请参见后续章节的说明。

原型

ErrorCode ***SendDataFile***(String filename)

参数

String filename: 如果未指定路径，则不使用当前文件夹中的命令定义文件。

解释说明

当执行 CloseMonPriner 时，已注册的命令数据被丢弃。如果发现同名的命令已被注册，则退出命令注册并返回一个错误。
可以注册的命令数只受系统的可用内存空间的限制。
调用 APD4 Status API 中的 BiSendDataFile。
有关详情，请参见第 60 页 “BiSendDataFile”。

DirectSendRead

通过 SendDataFile 传输已注册的命令，并接收由数据类型名称指定的数据。
调用 APD4 Status API 中的 BiDirectSendRead。有关详情，请参见第 62 页 “BiDirectSendRead”。

原型

1. ErrorCode ***DirectSendRead***(String cmdName, String readName, ref byte[] readBuf, int timeout, bool nullTerminate)
   描述 : 发送由 SendDataFile 定义的命令至 TM 打印机，并从打印机接收执行结果 （二进制数据）。
2. ErrorCode ***DirectSendRead***(String cmdName, String readName, out String response, int timeout)
   描述 : 发送由 SendDataFile 定义的命令至 TM 打印机，并从打印机接收执行结果 （字符串数据）。
3. ErrorCode ***DirectSendRead***(String cmdName, String readName, int timeout)
   描述 : 仅发送由 SendDataFile 定义的命令至 TM 打印机。不从打印机接收执行结果。

参数

String cmdName: 命令名称
String readName: 接收数据的数据类型名称
ref byte[] readBuf: 接收到的数据
int timeout: 数据传输和数据接收的超时时间 （以毫秒为单位）
bool nullTerminate: 当接收到 NULL 时判断是否结束接收
当指定 “False”时，读取由 readBuf 指定的数据或从 TM 打印机读取数据直至出现一个超时错误。
out String response: 接受到的数据

SetDefaultEchoTime

设置网络打印机回应确认次数的初始值和每次回应确认的超时时间。
当 APD4StatusAPI 第一次初始化时，回应确认次数是一次，每次回应确认的超时时间是一秒。
重启计算机后，通过 API 的配置将有效。
注意此 API 只有在 Ethernet 连接的情况下才能执行。
调用 APD4 Status API 中的 BiSetDefaultEchoTime。有关详情，请参见第 65 页“BiSetDefaultEchoTime”。

原型

ErrorCode ***SetDefaultEchoTime***(int count, int timeout)

参数

int count: 回应确认次数
int timeout: 每次回应确认的超时时间 （以毫秒为单位）

SetEtherEchoTime

设置网络打印机回应确认次数的值和每次回应确认的超时时间。在执行本 API 前，使用由 SetDefaultEchoTime 设置的值。注意此 API 只有在 Ethernet 连接的情况下执行才有效。
调用 APD4 Status API 中的 BiSetEtherEchoTime。有关详情，请参见第 66 页“BiSetEtherEchoTime”。

原型

ErrorCode ***SetEtherEchoTime***(int count, int timeout)

参数

int count: 回应确认次数
int timeout: 每次回应确认的超时时间

事件

StatusCallback

处理 ASB 状态通知的事件。
对应于 APD4 Status API 中 BiSetStatusBackFunction 指定的回调函数。有关详情，请参见第 48 页 “BiSetStatusBackFunction”。

原型

StatusCallbackHandler(ASB asb)

参数

ASB asb: com.epson.pos.driver 中定义的 ASB statusConstants。
ASB 应该用于此值。

StatusCallbackEx

处理 ASB 状态通知的事件。
对应于 APD4 Status API 中 BiSetStatusBackFunctionEx 指定的回调函数。
有关详情，请参见第 49 页 “BiSetStatusBackFunctionEx”。

原型

StatusCallbackHandlerEx(ASB asb, String portName)

参数

ASB asb: com.epson.pos.driver 中定义的 ASB statusConstants。
ASB 应该用于此值。
String portName: 端口名称

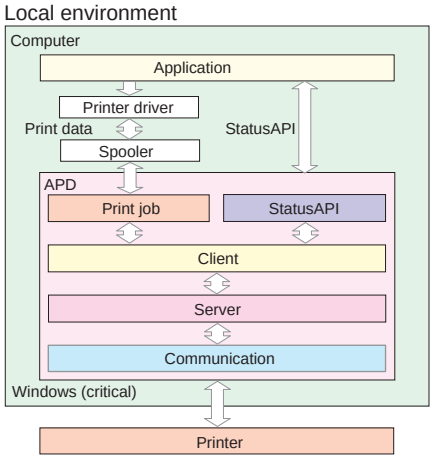
创建日志文件

本章解释说明了如何输出和查看日志文件。

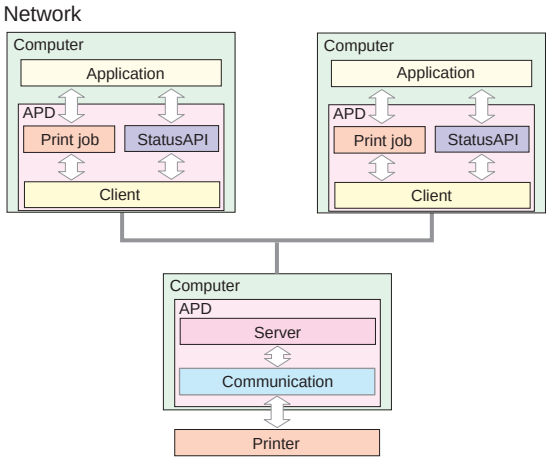
APD 允许创建一个日志文件，通过其可以帮助您很快地解决问题。日志文件以文件名 EpsonPOSPort.log 保存。其主要特征如下所示。

- 当打印系统出现问题，除了查看 APD 日志外还可以查看 Windows 错误信息。
- 可以获取一个进程 ID。当有多个进程被执行时，可以通过其鉴别哪个日志对应于哪个进程。
- 支持客户服务器系统。可以获取客户端 / 服务器端的日志。
- 对于以下各项均可以产生日志文件：Status API，打印任务，客户端，服务器端，以及通讯模块。

以下图表表示 APD 日志的获取位置。各模块的日志按照获取的顺序作为单个日志文件输出。



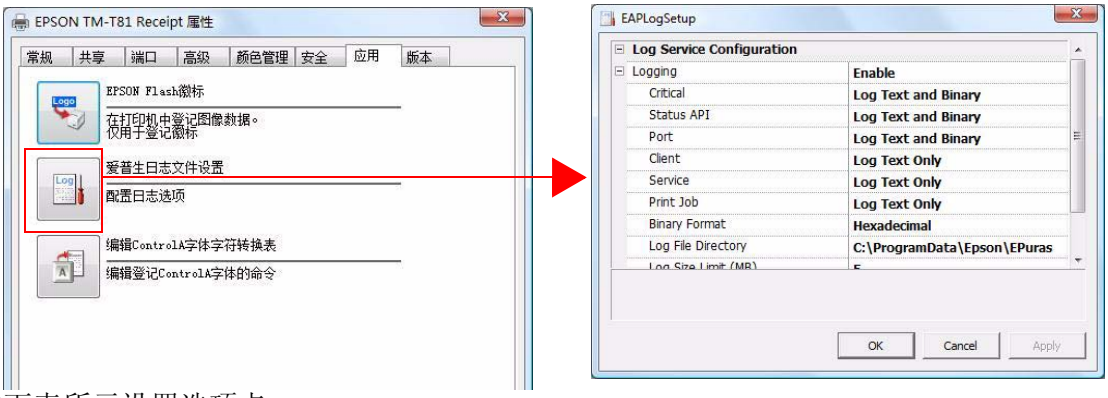
客户服务器系统的模块配置如下图所示。



每台计算机均产生一个日志文件。计算机的日期和时间设置提供给记时打印机。

日志文件设置

您可以选择是否输出日志文件，生成哪个模块的日志，和日志文件输出到哪里。
在属性屏面上选定应用选项，单击 [爱普生日志文件设置] 按钮。



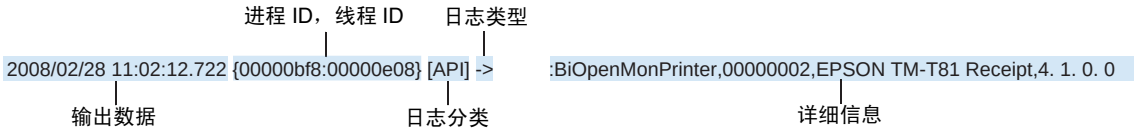
如下表所示设置选项卡。

设置		说明
Logging	Enable （默认）	启用日志输出。
	Disable	禁用日志输出。
Critical	选择 Windows 错误信息如何输出。	
	Log Text Only	以文本格式输出日志。
	Log Text and Binary （默认）	以文本格式和二进制数据输出格式。
Status API	选择 Status API 日志如何输出。	
	Do Not log	Status API 日志不输出。
	Log Text Only	以文本格式输出日志。
	Log Text and Binary （默认）	以文本格式和二进制数据输出格式。
Port	选择通讯端口的日志如何输出。	
	Do Not log	通讯端口日志不输出。
	Log Text Only	以文本格式输出日志。
	Log Text and Binary （默认）	以文本格式和二进制数据输出格式。
Client	选择客户 - 服务器端系统上应用程序的日志如何输出。	
	Do Not log	客户端日志不输出。
	Log Text Only （默认）	以文本格式输出日志。
	Log Text and Binary	以文本格式和二进制数据输出格式。
Service	选择客户 - 服务器端系统上服务器的日志如何输出。	
	Do Not log	服务器日志不输出。
	Log Text Only （默认）	以文本格式输出日志。
	Log Text and Binary	以文本格式和二进制数据输出格式。
Print Job	选择打印任务的日志如何输出。	
	Do Not log	打印任务信息日志不输出。
	Log Text Only （默认）	以文本格式输出日志。
	Log Text and Binary	以文本格式和二进制数据输出格式。

设置		说明
Binary Format	配置二进制数据格式。	
	Hexadecimal （默认）	输出十六进制日志文件。
	Base64	输出 Base64 日志文件。
Log File Directory	指定日志文件的输出位置。 （默认） Windows XP: C:\Documents and Settings\All Users\Application Data\Epson\EPuras Windows 7 / Vista: C:\ProgramData\Epson\EPuras	
Log Size Limit (MB)	指定日志文件大小的上限。 当超出上限，日志文件使用 zip 格式压缩，并作为备份文件保存。以后的日志信息被保存为一个新的日志文件。一组连续的号码被应用于 BAK 文件的命名。 （例如：EpsonPOSPort1.bak）。 指定需备份的日志文件数。 （范围：1 至 1024，默认：5）	
Backup File Count	指定需备份的日志文件数。 （范围 1 至 9，默认：1）	

查看日志文件

日志文件按照以下所示查看。



日志数据	解释说明
输出数据	YYYY/MM/DD hh:mm:ss.sss
进程 ID，线程 ID	{ 进程 ID，线程 ID}
日志分类	表明本日志文件对应哪个模块。 （请参见第 86 页 “日志分类”。）
日志类型	-> 调用函数 <- 返回函数 -- 执行函数 （调用，返回） ** 发生一个事件
详细信息	有关每个模块和日志类型的信息 当一个函数被执行： 函数名称称 （参数 1， ---，参数 n） < 返回 >[执行时间 （毫秒）]

日志分类

模块	日志分类	详细信息的内容
Critical	!!!	Windows 的重大事件和错误
Status API	API	Status API 调用函数及其参数信息，或被调用的 Status API 函数及其参数和返回信息
Port	PRT	端口控制，接口的事件细节，和输入 / 输出的数据信息
Client	CLI	客户 - 服务器系统上应用程序的进程信息。
Server	SVR	客户 - 服务器系统上服务器的进程信息
Print Job	SPL	端口打开 / 关闭信息以及输出到 / 输入到的端口信息

日志输出示例

```
2008/02/28 11:02:12.722 {00000bf8:00000e08} [API] ->    :BiOpenMonPrinter,00000001,EPSON TM-T81 Receipt,4. 1. 2. 0
2008/02/28 11:02:12.722 {00000bf8:00000e08} [CLI] -> Open('pipe://TM/ESDPRT001', 0x01473460)
2008/02/28 11:02:12.722 {000006dc:00000cd0} [SVR] -> 0036d4e8::Open(0, TM/ESDPRT001)
2008/02/28 11:02:12.722 {000006dc:00000cd0} [SVR] <- 0036d4e8::Open(8, TM/ESDPRT004) <00000000>
2008/02/28 11:02:12.722 {00000bf8:00000e08} [CLI] ** (TM/ESDPRT001)Event(0x00010003) 4:
2008/02/28 11:02:12.722 {000006dc:00000cdc} [SVR] -- 0036d4e8::RegisterCallback(8, 00010001) <PHR_SUCCESS>
2008/02/28 11:02:12.722 {00000bf8:00000e08} [CLI] <- Open('pipe://TM/ESDPRT001', 1) <00000000>
2008/02/28 11:02:12.722 {00000bf8:00000e08} [CLI] -> RegisterCallback(1, 0x00040002, 0x01396e00, 0x0177f2f0)
2008/02/28 11:02:12.722 {000006dc:00000c7c} [SVR] -- 0036d4e8::RegisterCallback(8, 00040002) <PHR_SUCCESS>
2008/02/28 11:02:12.722 {00000bf8:00000e08} [CLI] <- RegisterCallback(1, 0x00040002, 0x01396e00, 0x0177f2f0)
                        <00000000>
```

附录

机型信息

本文档解释说明了 TM-T81 的通过 Status API 获取的信息。

TM-T81

ASB 状态

宏定义	打开 / 关闭	值	状态
ASB_NO_RESPONSE	打开	0x00000001	无打印机回应
	关闭	0x00000000	打印机回应
ASB_PRINT_SUCCESS	打开	0x00000002	打印完成
	关闭	0x00000000	-
ASB_DRAWER_KICK	打开	0x00000004	3 号货币纸盒开启连接器插针的状态 = “H”
	关闭	0x00000000	3 号货币纸盒开启连接器插针的状态 = “L”
ASB_OFF_LINE	打开	0x00000008	脱机状态
	关闭	0x00000000	联机状态
ASB_COVER_OPEN	打开	0x00000020	盖板打开
	关闭	0x00000000	盖板关闭
ASB_PAPER_FEED	打开	0x00000040	进纸器开关正在进纸
	关闭	0x00000000	进纸器开关停止进纸
ASB_AUTOCUTTER_ERR	打开	0x00000800	发生自动裁纸器错误
	关闭	0x00000000	未发生自动裁纸器错误
ASB_UNRECOVER_ERR	打开	0x00002000	发生不可恢复的错误
	关闭	0x00000000	未发生不可恢复的错误
ASB_AUTORECOVER_ERR	打开	0x00004000	发生自动恢复的错误
	关闭	0x00000000	未发生自动恢复的错误
ASB_RECEIPT_NEAR_END	打开	0x00020000	接近卷纸末端传感器监测到无纸
	关闭	0x00000000	接近卷纸末端传感器监测到仍有纸

宏定义	打开 / 关闭	值	状态
ASB_RECEIPT_END	打开	0x00080000	卷纸末端传感器监测到无纸
	关闭	0x00000000	卷纸末端传感器监测到仍有纸
ASB_SPOOLER_IS_STOPPED	打开	0x80000000	停止卷轴
	关闭	0x00000000	开启卷轴

维护计数器

计数器编号 (readno)	可重置	计数器 （读取计数器）	单位
20 (14H)	可重置	进纸行数 （对于卷纸）	行
21 (15H)	可重置	启动打印头的次数 （对于卷纸）	次
50 (32H)	可重置	自动裁纸器的操作次数	次
70 (46H)	可重置	打印机的操作时间	小时
148 (94H)	累积	进纸行数 （对于卷纸）	行
149 (95H)	累积	启动打印头的次数 （对于卷纸）	次
178 (B2H)	累积	自动裁纸器的操作次数	次
198 (C6H)	累积	打印机的操作时间	小时