

# Advanced Printer Driver 6

# Status API 手册

---

## 使用之前

本章对使用本产品之前应了解的信息进行说明。

## APD6 概述

对 APD6 的概述进行说明。

## 使用 Status API

说明如何建立开发环境获取 ASB 状态以及如何处理 ASB 状态

## Win32 参考

解释说明在 Win32 环境下使用 Status API

## .NET 参考

解释说明在 .NET 环境下使用 Status API



# 使用之前

本章对使用 EPSON Advanced Printer Driver 6（以下简称 APD6）之前应了解的信息进行说明。

## APD6 的程序包

APD6 由以下程序包构成。


- 打印机驱动程序包  
是按 TM 打印机的机型准备的程序包。安装打印机驱动程序后，即可通过应用软件简单地进行打印。收录有以下手册。
  - 安装手册  
对 APD6 的安装、TM 打印机的注册、打印机驱动程序的自动安装方法等进行说明。
  - TM 打印机手册  
对 APD6 的使用方法与功能进行说明。
  - 打印机规格  
对各 TM 打印机机型的打印机驱动程序规格进行说明。
- StatusAPI 程序包  
是 APD6 专用的 TM 打印机通用程序包。开发使用 Status API 控制 TM 打印机的应用程序或同时使用 APD6 与其他 EPSON 驱动程序时需要安装。收录有以下手册。
  - Status API 手册（本书）  
对使用 Status API 并通过应用软件获取 TM 打印机状态的方法进行说明。有关各 TM 打印机机型可使用的 API 等的规格，请参阅打印机驱动程序包中的“打印机规格”手册。
- 示例程序包  
是 APD6 专用的 TM 打印机通用程序包。收录有用于开发 TM 打印机控制与打印应用程序的示例程序与源代码。没有手册，但收录有说明程序的 HTML 文件。

## 下载

可从以下网站下载打印机驱动程序、实用程序、手册类的最新版本。

[www.epson-biz.com/](http://www.epson-biz.com/)

## 标记的含义

|  |                 |
|--|-----------------|
|  参考 | 记载了补充说明或应了解的事项。 |
|--|-----------------|

# 目录

---

|                   |   |
|-------------------|---|
| 使用之前 .....        | 2 |
| ■ APD6 的程序包 ..... | 2 |
| 下载.....           | 2 |
| ■ 标记的含义 .....     | 3 |
| ■ 目录 .....        | 4 |

---

|                            |   |
|----------------------------|---|
| APD6 概述 .....              | 6 |
| ■ APD6 的特点 .....           | 6 |
| 术语表.....                   | 6 |
| ■ 运行环境 .....               | 7 |
| 需要安装 StatusAPI 程序包的软件..... | 7 |
| 支持 TM 打印机.....             | 7 |
| ■ 开发语言 .....               | 8 |
| ■ StatusAPI 程序包的安装 .....   | 8 |
| 卸载.....                    | 8 |

---

|                                    |    |
|------------------------------------|----|
| 使用 Status API .....                | 9  |
| ■ 开发环境的结构 .....                    | 9  |
| ■ Status API 函数的类型 .....           | 12 |
| ■ 编程流程 .....                       | 13 |
| Visual C++.....                    | 13 |
| Visual C# / Visual Basic .NET..... | 15 |
| ■ Status API 错误及对应方案.....          | 16 |
| ASB 状态.....                        | 16 |
| Status API 执行错误.....               | 17 |

---

|                             |    |
|-----------------------------|----|
| Win32 参考 .....              | 18 |
| ■ BiOpenMonPrinter .....    | 18 |
| ■ BiCloseMonPrinter .....   | 20 |
| ■ BiLockPrinter .....       | 21 |
| ■ BiUnlockPrinter .....     | 23 |
| ■ BiDirectIOEx .....        | 24 |
| ■ BiResetPrinter .....      | 27 |
| ■ BiForceResetPrinter ..... | 28 |
| ■ BiGetType .....           | 29 |

---

|                                   |    |
|-----------------------------------|----|
| ■ BiGetStatus .....               | 30 |
| ■ BiSetStatusBackFunction .....   | 31 |
| ■ BiSetStatusBackFunctionEx ..... | 32 |
| ■ BiCancelStatusBack .....        | 33 |
| ■ BiPowerOff .....                | 34 |
| ■ BiGetPrnCapability .....        | 35 |
| ■ BiOpenDrawer .....              | 36 |

---

## . NET 参考 ..... 38

|                        |    |
|------------------------|----|
| ■ 属性 .....             | 38 |
| IsValid.....           | 38 |
| LastError.....         | 38 |
| Status.....            | 38 |
| ■ 方法 .....             | 39 |
| OpenMonPrinter.....    | 39 |
| CloseMonPrinter.....   | 39 |
| LockPrinter.....       | 39 |
| UnlockPrinter.....     | 40 |
| DirectIOEx.....        | 40 |
| ResetPrinter.....      | 41 |
| ForceResetPrinter..... | 41 |
| GetType.....           | 41 |
| SetStatusBack.....     | 42 |
| CancelStatusBack.....  | 42 |
| PowerOff.....          | 42 |
| GetPrnCapability.....  | 43 |
| OpenDrawer.....        | 43 |
| ■ 事件 .....             | 44 |
| StatusCallback.....    | 44 |
| StatusCallbackEx.....  | 44 |

---

## 附录 ..... 45

|                          |    |
|--------------------------|----|
| ■ Acknowledgements ..... | 45 |
| ■ 注意事项 .....             | 46 |
| ■ 商标 .....               | 46 |

# APD6 概述

## APD6 的特点

- APD6 是爱普生 TM 打印机专用的 Windows 打印机驱动程序。
- 通过安装 StatusAPI 程序包，即可实现以下功能。
- 可与使用爱普生其他驱动程序（OPOS 等）的应用程序共享 TM 打印机。驱动程序会自动对应用程序进行独占控制，不需要通过应用程序控制。
  - 可使用 TM 打印机的 Status API，从应用程序获取 TM 打印机的机型信息、控制 TM 打印机、获取 TM 打印机状态。
  - 应用程序在 .NET 环境下也可以使用 TM 打印机的设备字体进行打印。

## 术语表

| 术语     | 解释说明   |
|--------|--|
| 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。 |

# 运行环境

请参见 APD6 的《安装手册》。

## 需要安装 StatusAPI 程序包的软件

- 以下情况时，请安装 Status API 程序包。
- 开发使用 Status API 控制 TM 打印机的应用程序时
  - 在同一计算机上使用 APD6 与以下某个 EPSON 软件时

| 软件名称                            | 支持版本               |
|---------------------------------|--------------------|
| EPSON Advanced Printer Driver 4 | Ver. 4.56 或更高版本    |
| EPSON Advanced Printer Driver 5 | Ver. 5.09 或更高版本    |
| EPSON OPOS ADK                  | Ver. 2.68 或更高版本    |
| EPSON OPOS ADK for .NET         | Ver. 1.11.20 或更高版本 |
| EPSON JavaPOS ADK               | Ver. 1.11.20 或更高版本 |
| EpsonNet SimpleViewer           | Ver. 2.30 或更高版本    |
| TM Virtual Port Driver          | Ver7.10a 或更高版本     |

## 支持 TM 打印机

已安装的 APD6 所支持的 TM 打印机。详情请参见已安装的打印机驱动程序包所带的《TM 打印机规格》。

## 开发语言

---

### Win32

- Visual Basic 6.0
- Visual C++

---

### .NET

- Visual Basic .NET
- Visual C#

## StatusAPI 程序包的安装

StatusAPI 程序包的安装方法如下。

- 1 双击 StatusAPI 程序包的安装程序（APD6\_StatusAPI\_x.exe）开始安装。
- 2 按照画面指示进行安装。
- 3 在安装打印机驱动程序包并注册打印机后，如已安装 StatusAPI 程序包，请执行打印以切换通讯端口。

至此 StatusAPI 程序包便安装完成。

### 卸载

请参见《安装手册》卸载 StatusAPI。不可单独卸载 StatusAPI 程序包。

# 使用 Status API

本章描述了使用 StatusAPI 的应用程序开发环境的构筑、编程方法。

## 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。

Visual C++: [p. 9](#)

Visual Basic .NET: [p. 10](#)

Visual C#: [p. 11](#)



参考

此处描述了使用 Visual Studio 2005 的方法。

### Visual C++


以下是使用 C++ 构建的开发环境的示例。

- 1 启动 Microsoft Visual C++，会显示解决方案资源管理器。
- 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）。  
Save as file  
32 位 OS: "C:\Programfiles\Epson\Advanced Printer Tool\StatusAPI"  
64 位 OS: "C:\Program Files(x86)\Epson\Advanced Printer Tool\StatusAPI"
- 3 打开源文件。使用 #include 标识定义 EpsStmApi.h。  
定义方法: `#include "EpsStmApi.h"`
- 4 Visual C++ 环境就绪，以便开发使用 StatusAPI 的应用程序。

## 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 Basic .NET，会显示解决方案资源管理器。
- 2 在 Solution Explorer 中右击 [References]，并选择 [Add References]。

|  |  |
|--|--|
|  参考 | 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] 中指定如下内容。  
 32 位 OS: "C:\Programfiles\Epson\Advanced Printer Tool\StatusAPI"  
 64 位 OS: "C:\Program Files(x86)\Epson\Advanced Printer Tool\StatusAPI"
- 5 输入文件名 “EpsonStatusAPI.dll”，并点击 [OK]。
- 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 C#，会显示解决方案资源管理器。
- 2 在 Solution Explorer 中右击 [References]，并选择 [Add References]。




### 参考

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] 中指定如下内容。  
32 位 OS: "C:\Programfiles\Epson\Advanced Printer Tool\StatusAPI"  
64 位 OS: "C:\Program Files(x86)\Epson\Advanced Printer Tool\StatusAPI"
- 5 输入文件名 “EpsonStatusAPI.dll”，并点击 [OK]。
- 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 18 页“Win32 参考”。对于不同的打印机机型被支持的函数也不同。

|  |  |
|--|--|
|  参考 | 不同型号的打印机支持不同函数。<br>有关各个机型的详细信息，请参考“打印机规格”。 |
|--|--|

| 应用程序                  | 函数                        | 说明   |
|-----------------------|---------------------------|--|
| 开始 / 关闭<br>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。<br>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。<br>同时获取端口号。   |
|                       | BiCancelStatusBack        | 退出自动状态通知功能。此功能可用于 BiSetStatusBackFunction 和 BiSetStatusBackFunctionEx。   |
| 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。  |
| 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。   |
| 电源关闭预处理               | BiPowerOff                | 设置电源关闭或待机模式。<br>进行以下操作： <ul style="list-style-type: none"> <li>● 将界面设为 BUSY。</li> <li>● 将打印机设为待机模式。</li> </ul> |
| 发送 ESC/POS 命令         | BiDirectIOEx              | 可以发送和接收 ESC/POS 命令。不添加 ASB 禁止命令。   |

# 编程流程

此处通过序列图说明了使用 StatusAPI 的 ASB 状态的获取方法。

 参考

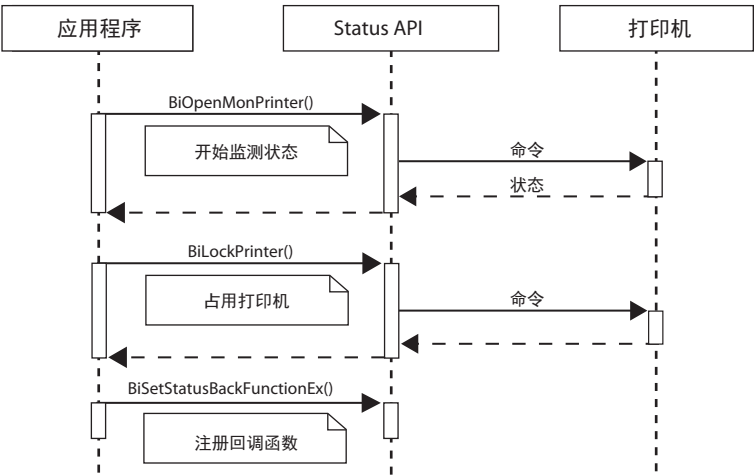
序列图省略了 DLL 的加载。

开始运行 Status API (BiOpenMonPrinter)，则 TM 打印机在每次状态变化时自动向 Status API 发送 ASB 状态。使用以下 API 获取被发送的 ASB 状态。

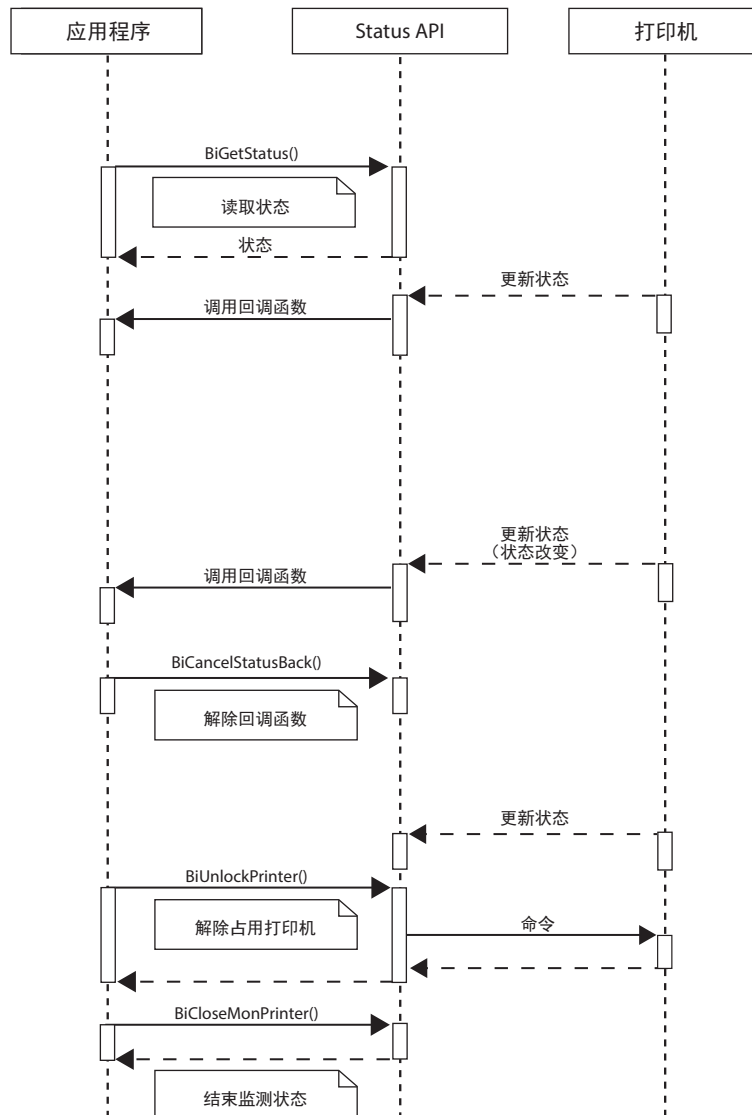
| Status API                | 说明  |
|---------------------------|---|
| BiGetStatus               | 用户（或应用程序）需要时获取 ASB 状态的 API。                   |
| BiSetStatusBackFunction   | 调用回调函数，向应用程序通知最新 ASB 状态的 API。                 |
| BiSetStatusBackFunctionEx | 调用回调函数，向应用程序通知最新 ASB 状态的 API。此外，还通知被回调的打印机端口。 |

## Visual C++

(1/2)



(2/2)

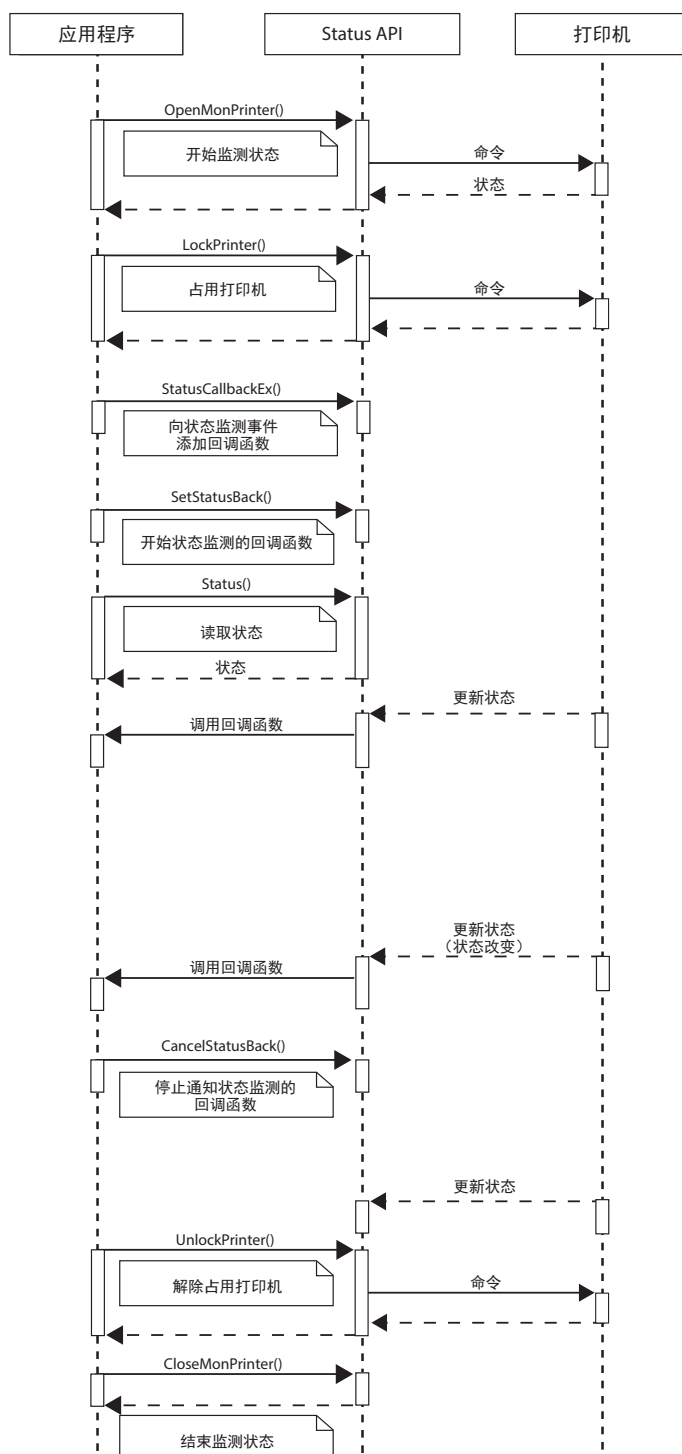


## Visual C# / Visual Basic .NET



参考

开发环境为 Visual C# 或 Visual Basic .NET 时，使用 .NET API。有关 .NET API 的详情，请参见第 38 页“[.NET 参考](#)”。



## 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，请参见“打印机规格”。

| 宏定义（常量）              | 原因   | 对应方案   |
|----------------------|--|--|
| ASB_NO_RESPONSE      | 打印机电源未开启。<br>通讯电缆未连接。<br>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。<br>通知在客户显示器上显示成功。<br>如果显示失败，不予以通知。 | -  |
| 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_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_PAPER_FEED       | 进纸。  | 进纸时没有问题。   |
| ASB_PANEL_SWITCH     | 正在按下打印机的面板开关。  | <ul style="list-style-type: none"> <li>按住打印机面板上的开关不会出现任何问题。</li> <li>通过事先设置为无效（FEED 按钮）的 TM 打印机的开关或按钮，例如，以 FEED 按钮的操作作为触发，也可以在应用程序中发生事件 / 操作。</li> </ul> |
| ASB_MECHANICAL_ERR   | 出现一个机械错误，即起始位置检测错误等。                                 | 排除错误原因，并再次接通 TM 打印机的电源或发送重置 TM 打印机的命令（BiResetPrinter 或 BiForceResetPrinter）。*  |
| ASB_AUTOCUTTER_ERR   | 发生了一个自动裁纸器错误。  | 排除错误原因，并再次接通 TM 打印机的电源或发送重置 TM 打印机的命令（BiResetPrinter 或 BiForceResetPrinter）。*  |
| ASB_DRAWER_KICK      | 货币纸盒打开。  | 刻意打开货币纸盒时没有问题。   |
| ASB_RECEIPT_END      | 缺纸。  | 更换打印纸。   |
| ASB_RECEIPT_NEAR_END | 打印纸将尽。   | 更换打印纸。   |
| ASB_COVER_OPEN       | 盖板打开。  | 关闭打印机盖板。   |
| ASB_WAIT_ON_LINE     | 等待返回在线状态。  | -  |

\* 详细说明请参照各打印机的操作手册。



参考

当创建开发环境时，使用 EpsStmApi.h 或者 Module1.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。<br>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。<br>（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。<br>例如：<br>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<br>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 服务是否已经启动。（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。                                     |



参考

当创建开发环境时，使用 EpsStmApi.h 或者 Module1.bas 文件进行宏定义。

# Win32 参考

本章解释说明了 Win32 环境下使用的 Status API 和语法。

 参考

- 数据类型在 C++ 中进行描述。
- 有关每台 TM 打印机可用的 API 和 ASB 状态、油墨状态、脱机原因、维护计数器，请参见 “打印机规格”。

## 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

```
int BiOpenMonPrinter (int nType, LPSTR pName)
```

#### 参数

nType:                    指定 pName 类型。

| 宏定义（常量）      | 值 | 说明                       |
|--------------|---|--------------------------|
| TYPE_PORT    | 1 | 指定 <i>pName</i> 中的端口名称。  |
| TYPE_PRINTER | 2 | 指定 <i>pName</i> 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-T88V Receipt”）。


例如）

- 由端口名 (ESDPRT001) 指定打印机。  
`nHandle = BiOpenMonPrinter(1, "ESDPRT001");`
- 由打印机名 (EPSON TM-T88V Receipt) 指定打印机。  
`nHandle = BiOpenMonPrinter(2, "EPSON TM-T88V Receipt");`
- 指定主机名 (SERVER) 的共享打印机 (EPSON TM-T88V Receipt)。  
`nHandle = BiOpenMonPrinter( 2, \\SERVER\\"EPSON TM-T88V Receipt");`

返回值

返回在 INT 类型中定义的变量。如成功使用 Status API，则返回识别打印机的句柄。即使打印机处于脱机状态，同样会返回这一句柄。返回下列状态 API 执行错误（值）。

| 宏定义（常量）           | 值    | 说明  |
|-------------------|------|---|
| ERR_TYPE          | -10  | <i>nType</i> 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 客户端和服务端之间的通讯错误。 |
| ERR_SPL_NOT_EXIST | -350 | 卷轴停止工作。                                       |

|  |   |
|--|---|
|  参考 | 有关 Status API 执行错误的恢复信息，请参照第 17 页“Status API 执行错误”。 |
|--|---|


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。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。                          |
| 脱机          | 返回该句柄。但打印机无法在脱机状态下打印时，切换至在线状态。 |
| 电缆移除 / 电源关闭 | 返回“ERR_ACCESS”。                |

# BiCloseMonPrinter

退出对打印机的监测状态。

|  |   |
|--|---|
|  参考 | 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。 |
|--|---|

## 语法


```
int BiCloseMonPrinter (int nHandle)
```

### 参数

nHandle: 指定句柄。

### 返回值

| 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 客户端和服务端之间的通讯错误。 |

|  |  |
|--|--|
|  参考 | 有关 Status API 执行错误的恢复信息，请参照第 17 页 “Status API 执行错误”。 |
|--|--|

# BiLockPrinter

锁定打印机。

 参考

- 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


```
int BiLockPrinter (int nHandle, DWORD timeout)
```

## 参数

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

## 返回值

| 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 客户端和服务端之间的通讯错误。 |
| ERR_RESET         | -400  | 打印机正在重启，无法调用                                  |
| ERR_LOCKED        | -1000 | 打印机被锁定。                                       |

 参考

有关 Status API 执行错误的恢复信息，请参照第 17 页 “Status API 执行错误”。

---

## 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]

- 客户端和服务端之间的连接断开。
- 客户端计算机进入待机或休眠模式。

# BiUnlockPrinter

解锁打印机。

|  |  |
|--|--|
|  参考 | <ul style="list-style-type: none"><li>• 本 API 用于共享打印机。</li><li>• 当使用本地打印机时，本 API 用于控制多个进程。</li></ul> |
|--|--|

## 语法


```
int BiUnlockPrinter (int nHandle)
```

### 参数

nHandle: 指定句柄。

### 返回值

| 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 客户端和服务端之间的通讯错误。 |
| ERR_RESET         | -400  | 打印机正在重启，无法调用                                  |
| ERR_LOCKED        | -1000 | 打印机被锁定。                                       |

|  |  |
|--|--|
|  参考 | 有关 Status API 执行错误的恢复信息，请参照第 17 页 “Status API 执行错误”。 |
|--|--|

## 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。

# 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。

 参考

- 请参阅以下网址的 ESC/ POS 命令的详细信息。  
[https://reference.epson-biz.com/modules/ref\\_escpos/index.php?content\\_id=2](https://reference.epson-biz.com/modules/ref_escpos/index.php?content_id=2)


## 语法

```
int BiDirectIOEx (int nHandle, DWORD writeLen, LPBYTE writeCmd,
                  LPDWORD readLen, LPBYTE readBuff, DWORD timeout,
                  BOOL nullTerminate, BYTE option)
```

## 参数

- nHandle: 指定句柄。
- writeLen: 指定写入打印机的数据长度。当为“0”时则不写入打印机。
- writeCmd: 指定写入打印机的数据（ESC/POS 命令）。
- 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”。
- readBuff: 指定保存从打印机读取的数据的缓冲。
- timeout: 以 ms（毫秒）为单位指定超时时间。
- 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。


| 值 | 说明               |
|---|------------------|
| 0 | 不获取 ASB 状态。      |
| 1 | 获取数据后，获取 ASB 状态。 |

 参考

- 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。
- 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

| 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，<br>所以此函数不可用。            |
| ERR_PH_NOT_EXIST     | -340  | PortHandler 不运行，或 PortHandler 客户端和服务端之间的通讯<br>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。   |

|  |  |
|--|--|
|  参考 | 有关 Status API 执行错误的恢复信息，请参照第 17 页 “Status API 执行错误”。 |
|--|--|

注释说明

通过确认返回值的返回值确认已正确执行该函数，或通过确认打印机操作确认已正确执行该命令。如果从打印机获取执行结果（指定 readLen），请确认执行结果。  
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          | 向返回值返回 “SUCCESS”。执行该命令。   |
| 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”。<br>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 | 向返回值返回 “ERR_ACCESS”。  |
| 打印中         | 向返回值返回 “ERR_LOCKED”。  |


注意事项

- 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<br>不更新<br>状态 |
|              | 否      | –              | 不回调<br>不更新<br>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。

 参考

- 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


```
int BiResetPrinter (int nHandle)
```

### 参数

nHandle: 指定句柄。

### 返回值


| 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 客户端和服务端之间的通讯错误。 |
| ERR_RESET         | -400  | 打印机正在重启，无法调用                                  |
| ERR_LOCKED        | -1000 | 打印机被锁定。                                       |

 参考

有关 Status API 执行错误的恢复信息，请参照第 17 页“Status API 执行错误”。

## 注释说明

通过确认返回值的返回值或通过重置打印机并确认打印机处于在线状态（确认 ASB 状态）来确认已正确执行该函数。

 参考

本函数执行后，15 秒内打印机无法接收打印命令。如果在这段时间执行打印，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          | 向返回值返回“SUCCESS”并进行重置。               |
| 脱机          | 向返回值返回“SUCCESS”并进行重置。               |
| 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 / 在多进程 / 在多用户的环境下，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。

 参考

- 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


```
int BiForceResetPrinter (int nHandle)
```

### 参数

nHandle:            指定句柄。

### 返回值


| 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 客户端和服务端之间的通讯错误。 |

 参考

有关 Status API 执行错误的恢复信息，请参照第 17 页 “[Status API 执行错误](#)”。

# BiGetType

获取打印机的类型 ID。

|  |                         |
|--|-------------------------|
|  参考 | 有关可获取的类型 ID 的信息，请联系经销商。 |
|--|-------------------------|

## 语法


```
int BiGetType (int nHandle, LPBYTE typeID, LPBYTE font,
               LPBYTE exrom, LPBYTE special)
```

## 参数

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

## 返回值

| 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 客户端和服务端之间的通讯错误。 |
| ERR_RESET         | -400  | 打印机正在重启，无法调用                                  |
| ERR_LOCKED        | -1000 | 打印机被锁定。                                       |

|  |  |
|--|--|
|  参考 | 有关 Status API 执行错误的恢复信息，请参照第 17 页 “Status API 执行错误”。 |
|--|--|

## 注释说明

由于机型的不同可能有无法获取的信息。此时将向 typeID 返回 0。

# BiGetStatus

获取当前打印机状态（ASB 状态）。

## 语法


```
int BiGetStatus (int nHandle, LPDWORD lpStatus)
```

## 参数

- nHandle: 指定句柄。
- lpStatus: 返回保存至 Status API 的 ASB 状态。ASB 状态由 4 个字节构成。

## 返回值

| 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 |

 参考

有关 Status API 执行错误的恢复信息，请参照第 17 页“[Status API 执行错误](#)”。

## 注释说明

有关每种打印机可以获取的 ASB 状态，请参考“打印机规格”。

# 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）。

## 语法

```
int BiSetStatusBackFunction
    (int nHandle,
     int (CALLBACK EXPORT *pFunction)
     (DWORD dwStatus))
```

## 参数


- nHandle: 指定句柄。
- \*pFunction: 指定回调函数的定义地址。

## 回调函数的参数

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

## 返回值


| 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 |

 参考

有关 Status API 执行错误的恢复信息，请参照第 17 页“[Status API 执行错误](#)”。

## 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。  
有关每种打印机可以获取的 ASB 状态，请参考“打印机规格”。

 参考

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。

## 语法

```
int BiSetStatusBackFunctionEx
    (int nHandle,
     int (CALLBACK EXPORT *pFunction)
     (DWORD dwStatus, LPSTR lpcPortName))
```

### 参数


- nHandle: 指定句柄。
- \*pFunction: 指定回调函数的定义地址。

### 回调函数的参数

- dwStatus: 返回保存至 Status API 的 ASB 状态。ASB 状态由 4 个字节构成。
- lpcPortName: 返回启动回调的打印机端口名称。

### 返回值


| 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 |

 参考

有关 Status API 执行错误的恢复信息，请参照第 17 页 “Status API 执行错误”。

## 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。  
有关每种打印机可以获取的 ASB 状态，请参考 “打印机规格”。

 参考

在注册的回调函数内不可使用 Status API。

# BiCancelStatusBack

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

## 语法


```
int BiCancelStatusBack (int nHandle)
```

## 参数

nHandle: 指定句柄。

## 返回值


| 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 |

 参考

- 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 17 页 “Status API 执行错误”。

# BiPowerOff

关闭 TM 打印机的电源或使其进入待机状态。

|  |   |
|--|---|
|  参考 | 本 API 在 TM 打印机处于脱机待恢复状态时无法调用。如果在本 API 的执行过程中 TM 打印机变为脱机状态，也将无法执行。 |
|--|---|

## 语法


```
int BiPowerOff (int nHandle)
```

### 参数

nHandle: 指定句柄。


### 返回值

| 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 客户端和服务端之间的通讯错误。 |
| ERR_LOCKED        | -1000 | 打印机被锁定。                                       |

|  |  |
|--|--|
|  参考 | 有关 Status API 执行错误的恢复信息，请参照第 17 页 “Status API 执行错误”。 |
|--|--|

# BiGetPrnCapability

通过打印机 ID 获取指定的打印机信息。



参考

- 有关可获取的打印机容量的信息，请联系经销商。
- 打印机不支持部分打印机 ID。如果指定不支持的打印机 ID，则会出现超时错误。

## 语法


```
int BiGetPrnCapability (int nHandle, BYTE prnID,
                        LPBYTE pBuffSize, LPBYTE pBuff)
```

## 参数

- nHandle:
- 指定句柄。
- prnID:
- 指定待获取打印机信息。
- 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。
- pBuff:
- 指定内存地址来设置打印机信息。

## 返回值

| 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 客户端和服务端之间的通讯错误。 |
| ERR_RESET            | -400  | 打印机正在重启，无法调用                                  |
| ERR_LOCKED           | -1000 | 打印机被锁定。                                       |




参考

有关 Status API 执行错误的恢复信息，请参照第 17 页 “Status API 执行错误”。

# BiOpenDrawer

打开货币纸盒。

|  |                   |
|--|-------------------|
|  参考 | 即使当打印机脱机时，打开货币纸盒。 |
|--|-------------------|

## 语法

```
int BiOpenDrawer (int nHandle, BYTE drawer, BYTE pulse)
```

### 参数

- nHandle: 指定句柄。
- drawer: 指定要打开的货币纸盒。


| 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: 指定何时货币纸盒开启信号为开。

| 宏定义（常量）          | 值 | 说明        |
|------------------|---|-----------|
| EPS_BI_PLUSE_100 | 1 | 100 毫秒的信号 |
| EPS_BI_PLUSE_200 | 2 | 200 毫秒的信号 |
| EPS_BI_PLUSE_300 | 3 | 300 毫秒的信号 |
| EPS_BI_PLUSE_400 | 4 | 400 毫秒的信号 |
| EPS_BI_PLUSE_500 | 5 | 500 毫秒的信号 |
| EPS_BI_PLUSE_600 | 6 | 600 毫秒的信号 |
| EPS_BI_PLUSE_700 | 7 | 700 毫秒的信号 |
| EPS_BI_PLUSE_800 | 8 | 800 毫秒的信号 |

### 返回值

| 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 客户端和服务端之间的通讯错误。 |
| ERR_RESET         | -400  | 打印机正在重启，无法调用                                  |
| ERR_LOCKED        | -1000 | 打印机被锁定。                                       |

|  |  |
|--|--|
|  参考 | 有关 Status API 执行错误的恢复信息，请参照第 17 页 “Status API 执行错误”。 |
|--|--|

---

## 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          | 向返回值返回 “SUCCESS”。打开货币纸盒。     |
| 脱机          | 向返回值返回 “SUCCESS”。打开货币纸盒。     |
| 电缆移除 / 电源关闭 | 向返回值返回 “ERR_ACCESS”。不打开货币纸盒。 |

# .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 17 页“Status API 执行错误”。

### Status

获取当前的打印机状态。

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 30 页“BiGetStatus”。

## 方法

### OpenMonPrinter

开始控制指定的打印机。有关详情，请参见[第 18 页](#) “[BiOpenMonPrinter](#)”。

#### 语法

```
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。

### CloseMonPrinter

停止控制指定的打印机。有关详情，请参见[第 20 页](#) “[BiCloseMonPrinter](#)”。

#### 语法

```
ErrorCode CloseMonPrinter ()
```

### LockPrinter

占用打印机。有关详情，请参见[第 21 页](#) “[BiLockPrinter](#)”。

#### 语法

```
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。有关详情，请参见第 23 页“[BiUnlockPrinter](#)”。

### 语法

```
ErrorCode UnlockPrinter ()
```

## 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。有关详情，请参见第 24 页“[BiDirectIOEx](#)”。

### 语法

- 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）。

- 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）。

- 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。<br>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。有关详情，请参见[第 27 页“BiResetPrinter”](#)。

---

### 语法

```
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。有关详情，请参见[第 28 页“BiForceResetPrinter”](#)。

---

### 语法

```
ErrorCode ForceResetPrinter()
```

## 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。有关详情，请参见[第 29 页“BiGetType”](#)。

---

### 语法

```
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    |

## SetStatusBack

通过 StatusCallback/StatusCallbackEx 事件开始状态通知。  
有关详情，请参见第 31 页 [“BiSetStatusBackFunction”](#)。

---

### 语法

```
ErrorCode SetStatusBack ()
```

## CancelStatusBack

通过 StatusCallback/StatusCallbackEx 事件停止状态通知。  
有关详情，请参见第 33 页 [“BiCancelStatusBack”](#)。

---

### 语法

```
ErrorCode CancelStatusBack ()
```

## PowerOff

执行打印机的电源关闭进程。  
有关详情，请参见第 34 页 [“BiPowerOff”](#)。

---

### 语法

```
ErrorCode PowerOff ()
```

## GetPrnCapability

获取由打印机 ID 指定的打印机的信息。有关详情，请参见第 35 页“[BiGetPrnCapability](#)”。

### 语法

- 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）。
- 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。有关详情，请参见第 36 页“[BiOpenDrawer](#)”。

### 语法

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:   | 指定何时货币纸盒开启信号为开。<br>com.epson.pos.driver.Pulse 中定义的常数应该用于获取该值。   |

## 事件

### StatusCallback

处理 ASB 状态通知的事件。有关详情，请参见第 31 页 [“BiSetStatusBackFunction”](#)。

---

#### 语法

*StatusCallbackHandler* (ASB asb)

#### 参数

ASB asb: ASB 状态。有关详情，请参见 “打印机规格”。

### StatusCallbackEx

处理 ASB 状态通知的事件。有关详情，请参见第 32 页 [“BiSetStatusBackFunctionEx”](#)。

---

#### 语法

*StatusCallbackHandlerEx* (ASB asb, String portName)

#### 参数

ASB asb: ASB 状态。有关详情，请参见 “打印机规格”。

String portName: 端口名称

# 附录

## Acknowledgements

---

### Info-ZIP

"Advanced Printer Driver" incorporate compression code from the Info-ZIP group.

-----  
This is version 2009-Jan-02 of the Info-ZIP license. The definitive version of this document should be available at <ftp://ftp.info-zip.org/pub/infozip/license.html> indefinitely and a copy at <http://www.info-zip.org/pub/infozip/license.html>.

Copyright (c) 1990-2009 Info-ZIP. All rights reserved.

For the purposes of this copyright and license, "Info-ZIP" is defined as the following set of individuals:

Mark Adler, John Bush, Karl Davis, Harald Denker, Jean-Michel Dubois, Jean-loup Gailly, Hunter Goatley, Ed Gordon, Ian Gorman, Chris Herborth, Dirk Haase, Greg Hartwig, Robert Heath, Jonathan Hudson, Paul Kienitz, David Kirschbaum, Johnny Lee, Onno van der Linden, Igor Mandrichenko, Steve P. Miller, Sergio Monesi, Keith Owens, George Petrov, Greg Roelofs, Kai Uwe Rommel, Steve Salisbury, Dave Smith, Steven M. Schweda, Christian Spieler, Cosmin Truta, Antoine Verheijen, Paul von Behren, Rich Wales, Mike White.

This software is provided "as is," without warranty of any kind, express or implied. In no event shall Info-ZIP or its contributors be held liable for any direct, indirect, incidental, special or consequential damages arising out of the use of or inability to use this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the above disclaimer and the following restrictions:

1. Redistributions of source code (in whole or in part) must retain the above copyright notice, definition, disclaimer, and this list of conditions.
2. Redistributions in binary form (compiled executables and libraries) must reproduce the above copyright notice, definition, disclaimer, and this list of conditions in documentation and/or other materials provided with the distribution. Additional documentation is not needed for executables where a command line license option provides these and a note regarding this option is in the executable's startup banner. The sole exception to this condition is redistribution of a standard UnZipSFX binary (including SFXWiz) as part of a self-extracting archive; that is permitted without inclusion of this license, as long as the normal SFX banner has not been removed from the binary or disabled.
3. Altered versions—including, but not limited to, ports to new operating systems, existing ports with new graphical interfaces, versions with modified or added functionality, and dynamic, shared, or static library versions not from Info-ZIP—must be plainly marked as such and must not be misrepresented as being the original source or, if binaries, compiled from the original source. Such altered versions also must not be misrepresented as being Info-ZIP releases—including, but not limited to, labeling of the altered versions with the names "Info-ZIP" (or any variation thereof, including, but not limited to, different capitalizations), "Pocket UnZip," "WiZ" or "MacZip" without the explicit permission of Info-ZIP. Such altered versions are further prohibited from misrepresentative use of the Zip-Bugs or Info-ZIP e-mail addresses or the Info-ZIP URL(s), such as to imply Info-ZIP will provide support for the altered versions.
4. Info-ZIP retains the right to use the names "Info-ZIP," "Zip," "UnZip," "UnZipSFX," "WiZ," "Pocket UnZip," "Pocket Zip," and "MacZip" for its own source and binary releases.

---

### Windows Template Library

使用 Microsoft 公司的 Windows Template Library。

## 注意事项

- (1) 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。
- (2) 本文档的内容如有变更，恕不另行通知。
- (3) 在准备此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。
- (4) 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。
- (5) 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。
- (6) 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和易耗件产品而导致的损坏或问题。

## 商标

EPSON 是 Seiko Epson Corporation 的注册商标。

Exceed Your Vision 以及 ESC/POS 为 Seiko Epson Corporation 的注册商标或商标。

Microsoft® Windows® Visual Studio® Visual C++® Visual C#® 是美国 Microsoft Corporation 在美国与其他国家的商标或注册商标。

所有其他商标均为其各自所有者的财产，仅供识别之用。

### ESC/POS® 指令系统

EPSON ESC/POS 是 POS 打印机专用的指令系统。该系统包含已获得专利或正在申请专利的指令。

ESC/POS 兼容爱普生的大部分 POS 打印机和显示设备。

通过使用 ESC/POS，可在 POS 环境中降低主机的处理负载。ESC/POS 由一套高功能、高效率的指令组成，并且具有非常高的灵活性，便于将来升级。

©Seiko Epson Corporation 2019 - 2021.