

Advanced Printer Driver 6

Status APIガイド

ご使用の前に

本製品をご使用いただく前に知っておいていただきたい情報について説明しています。

APD6 の概要

APD6の概要について説明します。

Status API の使い方

開発環境の構築、ASBステータスの取得方法について説明します。

Win32 API リファレンス

Win32環境で使用するStatus APIの情報について説明します。

.NET API リファレンス

.NET環境で使用するStatus APIの情報について説明します。



ご使用の前に

本章では、EPSON Advanced Printer Driver 6 (以降 APD6) をご使用いただく前に知っておいていただきたい情報について説明しています。

APD6 のパッケージ


APD6 は、以下のパッケージで構成されています。

- プリンタードライバーパッケージ
TM プリンターの機種ごとに用意されるパッケージです。プリンタードライバーをインストールするとアプリケーションソフトから簡単に印刷することができます。以下のマニュアルを収録しています。
 - 導入ガイド
APD6 のインストール、TM プリンターの登録、プリンタードライバーを自動的にインストールする方法などを説明しています。
 - 設定ガイド
APD6 の使用方法と機能について説明しています。
 - プリンター仕様
TM プリンター機種ごとのプリンタードライバーの仕様を説明しています。
- StatusAPI パッケージ
APD6 専用の TM プリンター共通のパッケージです。Status API を使用して TM プリンターを制御するアプリケーションを開発したり、APD6 と他のエプソンドライバーと共存させたりする場合にインストールする必要があります。以下のマニュアルを収録しています。
 - Status API ガイド (本書)
Status API を使い、アプリケーションソフトから TM プリンターのステータスを取得する方法を説明しています。TM プリンター機種ごとに使用可能な API などの仕様は、プリンタードライバーパッケージに含まれる「プリンター仕様」マニュアルを参照してください。
- サンプルプログラムパッケージ
APD6 専用の TM プリンター共通のパッケージです。TM プリンターの制御・印刷するアプリケーションを開発するための、サンプルプログラムとソースコードを収録しています。マニュアルはありませんが、プログラムを説明した HTML ファイルを収録しています

最新版のダウンロード

本製品の最新版は、下記の URL からダウンロードできます。
www.epson.jp/support/

マークの意味

 参考	補足説明や知っておいていただきたいことを記載しています。
---	------------------------------

もくじ

ご使用の前に 2

- APD6 のパッケージ 2
 - 最新版のダウンロード 2
- マークの意味 3
- もくじ 4

APD6 の概要 5

- APD6 の特長 5
 - 用語説明 5
- 動作環境 6
 - StatusAPI パッケージの必要なソフトウェア ... 6
 - サポート TM プリンター 6
- 開発言語 7
- StatusAPI パッケージのインストール 7
 - アンインストール 7

Status API の使い方 8

- 環境の構築 8
- Status API の種類 11
- プログラミングフロー 12
 - Visual C++ 12
 - Visual C# / Visual Basic .NET 14
- Status API のエラーと対処方法 15
 - ASB ステータス 15
 - 戻り値 17

Win32 API リファレンス 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 API リファレンス 38

- プロパティ 38
 - IsValid 38
 - LastError 38
 - Status 39
- メソッド 40
 - OpenMonPrinter 40
 - CloseMonPrinter 40
 - LockPrinter 40
 - UnlockPrinter 41
 - DirectIOEx 41
 - ResetPrinter 42
 - ForceResetPrinter 42
 - GetType 42
 - SetStatusBack 43
 - CancelStatusBack 43
 - PowerOff 43
 - GetPrnCapability 44
 - OpenDrawer 44
- イベント 45
 - StatusCallback 45
 - StatusCallbackEx 45

付録 46

- Acknowledgements 46
- ご注意 47
- 商標 47

APD6 の概要

APD6 の特長

APD6 は、エプソン の TM プリンター専用の Windows プリンタードライバーです。

StatusAPI パッケージをインストールすることで、以下の機能を実現できます。

- エプソンの他のドライバー (OPOS など) を使用するアプリケーションと、TM プリンターを共有できます。アプリケーションの排他制御は、ドライバー側で自動的に行うので、アプリケーション側で制御する必要はありません。
- TM プリンターのステータス API を使用して、アプリケーションから TM プリンターの機種情報を取得したり、TM プリンターを制御したり、TM プリンターの状態を取得したりできます。
- .NET 環境のアプリケーションでも、TM プリンターのデバイスフォントを使用した印刷ができます。

用語説明

用語	説明
ASB ステータス	Auto Status Back の略で、TM プリンターの機能の一部です。TM プリンターの状態が変化 (カバー開閉 / 用紙なし / 印刷終了など) したときに、TM プリンターから自動的に送信されるステータスです。

動作環境

APD6 の「導入ガイド」を参照してください。

StatusAPI パッケージに必要なソフトウェア

Status API パッケージは以下の場合にインストールしてください。

- Status API を使用して TM プリンターを制御するアプリケーションを開発する場合
- 同じコンピューター上で APD6 と以下のエプソン製ソフトウェアのいずれかと使用する場合

ソフトウェア名	対応バージョン
EPSON Advanced Printer Driver 4	Ver.4.56 以降
EPSON Advanced Printer Driver 5	Ver.5.09 以降
EPSON OPOS ADK	Ver.2.68 以降
EPSON OPOS ADK for .NET	Ver.1.11.20 以降
EpsonNet SimpleViewer	Ver.2.30 以降
TM Virtual Port Driver	Ver7.10a 以降

サポート TM プリンター

インストールされている APD6 がサポートしている TM プリンターです。詳細は、インストールされているプリンタードライバーパッケージに同梱されている、「TM プリンター仕様」を参照してください。

開発言語

Win32

- Visual C++

.NET

- Visual Basic .NET
- Visual C#

StatusAPI パッケージのインストール

StatusAPI パッケージのインストール方法は以下のとおりです。

- 1** StatusAPIパッケージのインストーラー(APD6_StatusAPI_x.exe)をダブルクリックしてインストールを開始します。
- 2** 画面の指示に従ってインストールします。
- 3** プリンタードライバパッケージをインストールしてからプリンターを登録後、StatusAPI パッケージをインストールした場合は、通信ポートを切り替えるために印刷を実行してください。

以上で StatusAPI パッケージのインストールが完了しました。

アンインストール

StatusAPI のアンインストールは「導入ガイド」を参照してください。StatusAPI パッケージは、単独でアンインストールすることはありません。

Status API の使い方

本章では、Status API を使ったアプリケーション開発環境の構築、プログラミング方法を説明しています。

環境の構築

Status API を使ったアプリケーションの開発環境の構築は、開発ツールによって違います。

Visual C++: [8 ページ](#)

Visual Basic .NET: [9 ページ](#)

Visual C#: [10 ページ](#)



参考

ここでは、Visual Studio 2017 で説明します。

Visual C++

Visual C++ での開発環境構築例を示します。

- 1 Microsoft Visual C++ を起動すると、ソリューションエクスプローラーが表示されます。
- 2 Status API の定義ファイル (EpsStmApi.h) をコピーし、アプリケーションを開発する作業フォルダー（プロジェクトを作成したフォルダー）に貼り付けます。
 ファイルの保存先
 32 bit OS: "C:\Programfiles\Epson\Advanced Printer Tool\StatusAPI"
 64 bit OS: "C:\Program Files(x86)\Epson\Advanced Printer Tool\StatusAPI"
- 3 Source File を開きます。#include ディレクティブを使って EpsStmApi.h を定義します。
 定義方法: `#include "EpsStmApi.h"`
- 4 Visual C++ の環境が整います。Status API を使ったアプリケーションを開発できます。

Visual Basic .NET

Visual Basic .NET での開発環境構築例を示します。

- 1 Microsoft Visual Basic .NET を起動すると、ソリューションエクスプローラーが表示されます。
- 2 ソリューションエクスプローラーの[参照設定]を右クリックし、[参照の追加]を選択します。



参考

[参照設定] の項目が表示されない場合、ソリューションエクスプローラーの上の [すべてのファイルを表示] アイコンをクリックしてください。

- 3 「参照の追加」画面が表示されます。[参照] タブをクリックします。
- 4 [ファイルの場所] を以下にします。
32 bit OS: "C:\Programfiles\Epson\Advanced Printer Tool\StatusAPI"
64 bit OS: "C:\Program Files(x86)\Epson\Advanced Printer Tool\StatusAPI"
- 5 [ファイル名] に "EpsonStatusAPI.dll" を入力し、[OK] をクリックします。
- 6 ソリューションエクスプローラーの [参照設定]-[EpsonStatusAPI] を選択し、プロパティの [特定バージョン] を "False" にします。
- 7 ソースコードに、Imports ステートメントを使って、以下を記述してください。
Imports com.epson.pos.driver
- 8 Visual Basic .NET の環境が整います。Status API を使ったアプリケーションを開発できます。

Visual C#

Visual C# での開発環境構築例を示します。

- 1 Microsoft Visual C# を起動すると、ソリューションエクスプローラーが表示されます。
- 2 ソリューションエクスプローラーの[参照設定]を右クリックし、[参照の追加]を選択します。




参考

[参照設定] の項目が表示されない場合、ソリューションエクスプローラーの上の [すべてのファイルを表示] アイコンをクリックしてください。

- 3 「参照の追加」画面が表示されます。[参照] タブをクリックします。
- 4 [ファイルの場所] を以下にします。
32 bit OS: "C:¥Programfiles¥Epson¥Advanced Printer Tool¥StatusAPI"
64 bit OS: "C:¥Program Files(x86)¥Epson¥Advanced Printer Tool¥StatusAPI"
- 5 [ファイル名] に "EpsonStatusAPI.dll" を入力し、[OK] をクリックします。
- 6 ソリューションエクスプローラーの [参照設定]-[EpsonStatusAPI] を選択し、プロパティの [特定バージョン] を "False" にします。
- 7 ソースコードに、using キーワードを使って、以下を記述してください。
using com.epson.pos.driver
- 8 Visual C# の環境が整います。Status API を使ったアプリケーションを開発できます。

Status API の種類

以下の Status API が用意されています。Status API の詳細は、[18 ページ「Win32 API リファレンス」](#)を参照してください。

 参考	TM プリンターの機種によって使用できる Status API が異なります。 機種ごとに使用できる Status API、取得できる情報の詳細は、「プリンター仕様」を参照してください。
---	--

用途	Status API	説明
Status API の開始 / 終了	BiOpenMonPrinter	指定した TM プリンターで Status API の使用を開始します。
	BiCloseMonPrinter	Status API の使用を終了します。
TM プリンターを占有	BiLockPrinter	TM プリンターを占有します。 その間、他のプロセスからの API は受け付けません。
	BiUnlockPrinter	BiLockPrinter を解除します。
ASB ステータスの取得	BiGetStatus	アプリケーションの必要なときに、Status API から ASB ステータスを取得します。
	BiSetStatusBackFunction	Status API の ASB ステータス変化時に、自動的にアプリケーションに通知するコールバック関数の呼び出しを通知します。
	BiSetStatusBack FunctionEx	Status API の ASB ステータス変化時に、自動的にアプリケーションに通知するコールバック関数の呼び出しを通知します。ポート番号も取得します。
	BiCancelStatusBack	自動ステータス通知機能を解除します。 BiSetStatusBackFunction / BiSetStatusBack FunctionEx に適用されます。
TM プリンター情報の取得	BiGetType	TM プリンターのタイプ ID を取得します。 (マルチバイトコード文字の対応など)
	BiGetPrnCapability	TM プリンターのプリンター ID を取得します。 (機種名、ファームウェアのバージョンなど)
ドロアー制御	BiOpenDrawer	ドロアーをオープンします。
TM プリンターのリセット	BiResetPrinter	USB / Ethernet インターフェイス の TM プリンターをリセットします。シリアルインターフェイスの TM プリンターはリセットできません。
	BiForceResetPrinter	BiLockPrinter で占有されている TM プリンターもリセットできます。
電源オフの前処理	BiPowerOff	電源オフ状態もしくは待機状態になります。 以下の処理がされます。 インターフェイスを BUSY 状態にする 電源オフ 時の待機状態にする
ESC/POS コマンドを送る	BiDirectIOEx	ESC/POS コマンドを送受信できます。ASB ステータスの読み込みを制御できます。

プログラミングフロー

ここでは、Status API を使った ASB ステータスの取得方法を、シーケンス図で説明します。



参考

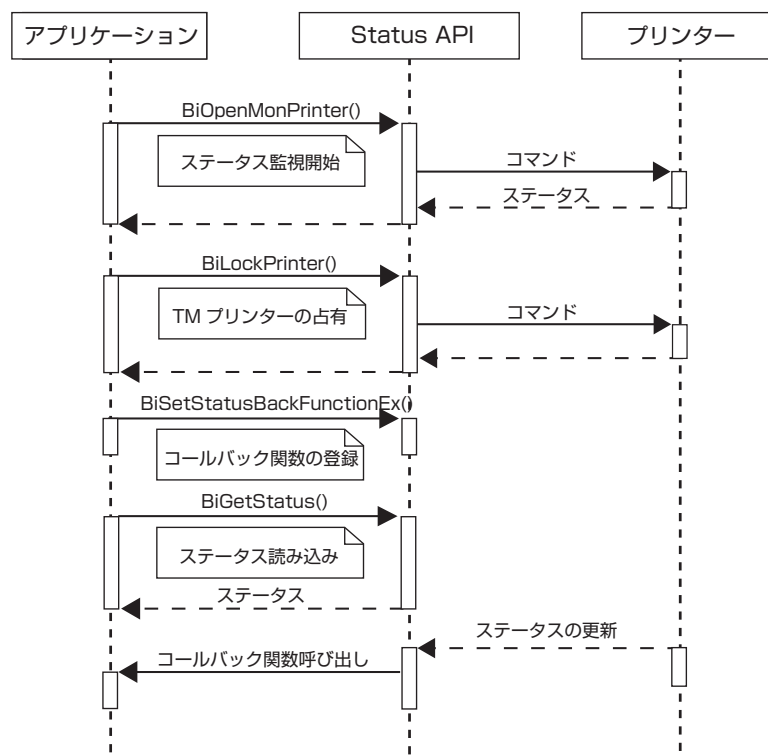
シーケンス図は、DLL のロードを省略しています。

Status API を開始 (BiOpenMonPrinter) すると、TM プリンターはステータスが変わるたびに ASB ステータスを、自動で Status API へ送信します。送信された ASB ステータスは、以下の API を使って取得します。

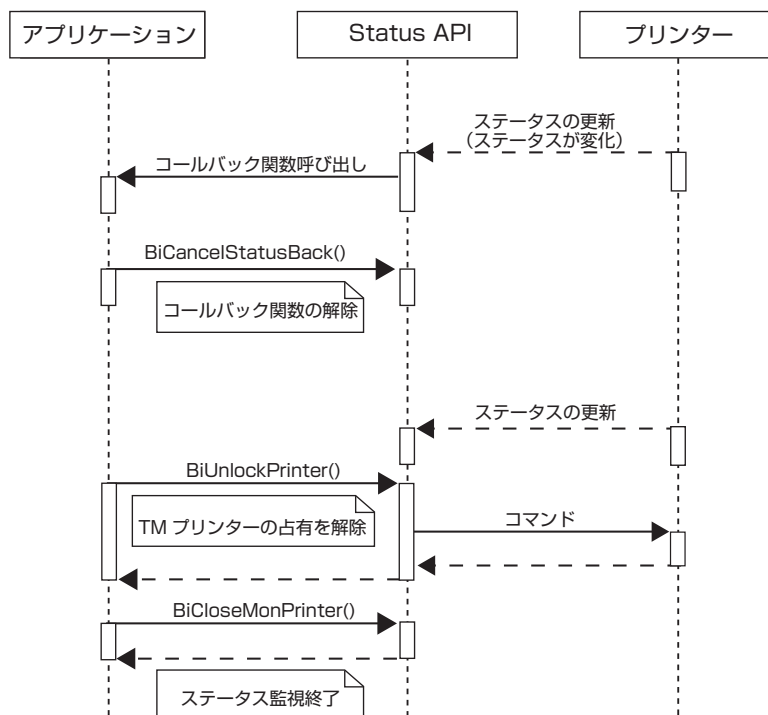
Status API	説明
BiGetStatus	ユーザー(またはアプリケーション)が必要なときに ASB ステータスを取得する API です。
BiSetStatusBack Function	コールバック関数を呼び出して、最新の ASB ステータスをアプリケーションに通知する API です。
BiSetStatusBack FunctionEx	コールバック関数を呼び出して、最新の ASB ステータスをアプリケーションに通知する API です。また、コールバックされたプリンターポートも通知されます。

Visual C++

(1/2)



(2/2)

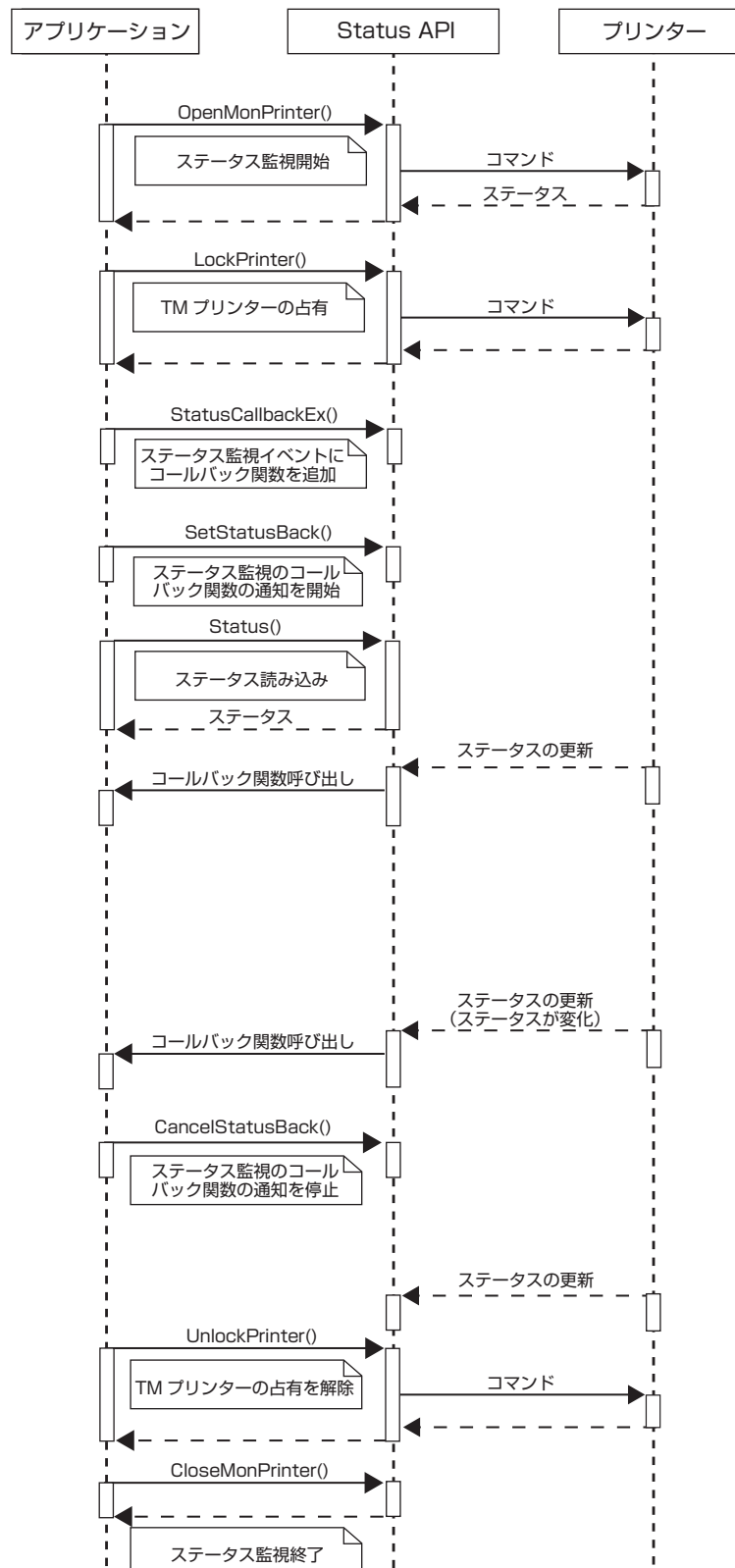


Visual C# / Visual Basic .NET



参考

開発環境が Visual C# や Visual Basic .NET の場合、.NET API を使用します。.NET API の詳細は、[38 ページ「.NET API リファレンス」](#)を参照してください。



Status API のエラーと対処方法

Status API のエラーには、ASB ステータスが通知するエラーと、Status API 呼び出し時に発生するエラーがあります。ここではエラーの内容と対処方法を説明しますので、ユーザーのアプリケーションでエラー処理を行ってください。


ASB ステータス

ASB ステータスを取得したときに返ってくるエラーです。TM プリンターの機種によって内容が異なります。詳細は「プリンター仕様」を参照してください。

マクロ定義 (定数)	原因	対応方法
ASB_NO_RESPONSE	<ul style="list-style-type: none"> TM プリンターの電源が入っていません。 通信ケーブルが抜かれています。 指定したプリンタードライバー名 / ポートが違います。 	電源、ケーブルなど TM プリンターの状態やポートを確認してください。
ASB_PRINT_SUCCESS	<p>印刷終了時に、印刷が成功したことを通知します。印刷に失敗した場合は通知されません。</p> <p>カスタマーディスプレイへの表示が成功したことを通知します。表示に失敗した場合は、通知されません。</p>	-
ASB_UNRECOVER_ERR	TM プリンターに印刷できないエラーが発生しました。	すぐにTMプリンターの電源を切ってください。*
ASB_AUTORECOVER_ERR	ヘッドの温度が上昇しました。	時間の経過により、ヘッドの温度が下降すれば自動的に解除されます。*
ASB_OFF_LINE	オフラインになる要因が発生しました。	オフラインになる要因を取り除いてください。
ASB_PRINTER_FEED	紙送りしています。	紙送りの最中であれば問題ありません。
ASB_PANEL_SWITCH	TM プリンターのスイッチ、またはボタンが押されています。	<ul style="list-style-type: none"> TM プリンターのスイッチ、またはボタンが押されていれば問題ありません。 TM プリンターのスイッチ、またはボタンを無効 (例: FEED ボタン) に設定しておくことで、たとえば FEED ボタン操作をトリガにアプリケーション側でイベント発生 / ハンドリングさせるといった使い方もできます。
ASB_MECHANICAL_ERR	ホームポジション検出エラーなどのメカニカルエラーが発生しました。	エラーの原因を取り除き、TM プリンターの電源を再投入または TM プリンターのリセットコマンド (BiResetPrinter または BiForceResetPrinter) を送ってください。*

マクロ定義 (定数)	原因	対応方法
ASB_AUTOCUTTER_ERR	オートカッターエラーが発生しました。	エラーの原因を取り除き、TM プリンターの電源を再投入または TM プリンターのリセットコマンド (BiResetPrinter または BiForceResetPrinter) を送ってください。*
ASB_DRAWER_KICK	ドロアーが開いています。	意図的に開けているのであれば、問題ありません。
ASB_RECEIPT_END	用紙がなくなりました。	TM プリンターに用紙を入れてください。
ASB_RECEIPT_NEAR_END	用紙残量が少なくなりました。	TM プリンターに用紙を入れてください。
ASB_COVER_OPEN	カバーが開いています。	TM プリンターのカバーを閉めてください。

*: 各 TM プリンターの詳細取扱説明書を参照してください。

 参考	マクロ定義は、開発環境を構築したときの、EpsStmApi.h ファイルおよび Module1.bas ファイルで定義されています。
---	--

戻り値

Status API の関数を呼び出したときに発生するエラー (Status API 関数の戻り値) です。Status API の関数によって内容が異なります。詳細は [18 ページ「Win32 API リファレンス」](#) を参照してください。

マクロ定義 (定数)	原因	対応方法
ERR_TYPE	nType のパラメーターが違っています。	正しい値を指定してください。
ERR_OPENED	指定された TM プリンターはすでにオープン済みです。	すでにオープンしているので、そのハンドル値を使用するか、別の TM プリンターを指定してください。
ERR_NO_PRINTER	指定されたプリンタードライバがありません。	プリンタードライバ名を確認してください。
ERR_NO_TARGET	指定された TM プリンターが見つかりません。 意図していない TM プリンターが接続されています。	正しい TM プリンターを接続してください。
ERR_NO_MEMORY	メモリ不足です。	使用できるメモリを追加してください。
ERR_HANDLE	TM プリンターを指定するハンドル値が不正です。	ハンドル値を確認してください。
ERR_TIMEOUT	タイムアウトエラーです。	このエラーが継続して発生する場合、TM プリンターが正しく接続されているか確認してください。
ERR_ACCESS	TM プリンターにアクセスできません。 (TM プリンターの電源が入っていない、ケーブルの接続不良等)	TM プリンターを確認してください。 (TM プリンターの電源、ケーブルの接続等)
ERR_PARAM	パラメーターエラーです。	パラメーターの指定が間違えています。パラメーターを見直してください。
ERR_NOT_SUPPORT	サポートしていない機種です。	サポートしていない機種では使用できません。
ERR_EXIST	指定されたデータがすでに存在しています。	存在しているデータを削除してください。 例：BiSetStatusBackXXX 実行時にエラー→BiCancelStatusBack 実行してから再実行
ERR_EXEC_FUNCTION	他のアプリケーションで Status API が使用されているため使用できません。	他のアプリケーションで使用している Status API を終了してください。
ERR_PH_NOT_EXIST	PortHandler が起動していません。 PortHandler に接続されていません。	PortHandler が正しく動作しているか確認してください。
ERR_SPL_NOT_EXIST	スプーラーサービスが起動していません。	Print Spooler サービスが開始されているか確認してください。 (Control Panel- 管理ツール - サービス)
ERR_RESET	リセット中のため使用できません。	少し待ってから再度呼び出してください。
ERR_LOCKED	TM プリンターがロックされています。	ロックが解除されるのを待つか、ロックしているアプリケーションで BiUnlockPrinter を実行してください。



参考

マクロ定義は、開発環境を構築したときの、EpsStmApi.h ファイルおよび Module1.bas ファイルで定義されています。

Win32 API リファレンス

本章では、Win32 環境で使用する Status API と構文について説明しています。



参考

- 本書では、データ型を Visual C++ で説明しています。
- 各 TM プリンターで利用できる API や ASB ステータス、オフライン要因については、「プリンター仕様」を参照してください。

BiOpenMonPrinter

指定された TM プリンターで Status API の使用を開始し、ハンドルを返します。
 複数のプロセスから 1 台の TM プリンターを同時にオープンすることができます。
 すでにオープンした TM プリンターを同一プロセスから再度オープンした場合、異なるハンドルが返ります。
 この場合は、どちらのハンドルも有効です。
 ただし、同一スレッドから 1 台の TM プリンターを同時にオープンすることはできません。

構文

int *BiOpenMonPrinter* (int nType, LPSTR pName)

パラメーター

nType: pName のタイプを指定します。

マクロ定義 (定数)	値	説明
TYPE_PORT	1	pName にポート名称を指定します。
TYPE_PRINTER	2	pName にプリンター名を指定します。

pName: nType で 1 を指定した場合、ポート名称 (例: "ESDPRT001") を指定します。
 nType を 2 で指定した場合、プリンター名 (例: "EPSON TM-T88V Receipt") を指定します。

例:

- ポート名 (ESDPRT001) から TM プリンターを指定する。
 nHandle = BiOpenMonPrinter(1, "ESDPRT001");
- プリンター名 (EPSON TM-T88V Receipt) から TM プリンターを指定する。
 nHandle = BiOpenMonPrinter(2, "EPSON TM-T88V Receipt");
- ホスト名 (SERVER) の共有プリンター (EPSON TM-T88V Receipt) を指定する。
 nHandle = BiOpenMonPrinter(2, "\\SERVER\` EPSON TM-T88V Receipt");

戻り値

INT 型で定義した変数に返します。Status API の使用開始に成功した場合、TM プリンターを識別するハンドルを返します。TM プリンターの状態がオフラインの場合でも、ハンドルを返します。オープンに失敗した場合、以下の戻り値を返します。

マクロ定義 (定数)	値	説明
ERR_TYPE	-10	nType のパラメーターエラー
ERR_OPENED	-20	指定された TM プリンターすでにオープン済み
ERR_NO_PRINTER	-30	指定された TM プリンターがない
ERR_NO_TARGET	-40	対象外の TM プリンター
ERR_NO_MEMORY	-50	メモリーが足りない
ERR_TIMEOUT	-70	タイムアウトエラー
ERR_ACCESS	-80	TM プリンターにアクセスできない (他アプリケーションがアクセス中もしくは、印刷中)
ERR_PARAM	-90	パラメーターエラー
ERR_PH_NOT_EXIST	-340	PortHandler が起動していない または、PortHandler のクライアント、サーバー間の通信エラー
ERR_SPL_NOT_EXIST	-350	スプーラーサービスが起動していない



参考

戻り値による対処方法は、[17 ページ「戻り値」](#)を参照してください。

説明

本 API は、他の Status API 関数を使用する前に呼び出します。戻り値のハンドルは、他の Status API 関数でパラメーターとして使います。一度に開始できる TM プリンターは、最大で 32 台です。

本 API で呼び出したときの、TM プリンターの状態によって以下の動作をします。

TM プリンターの状態	動作
オンライン	nHandle にハンドルを返します
オフライン	nHandle にハンドルを返しますが、この状態では印刷できないので、オンラインにする必要があります。
ケーブルが接続されていない / 電源オフ	戻り値に “ERR_ACCESS” を返します。

BiCloseMonPrinter

ステータスを監視している TM プリンターを解除します。

BiOpenMonPrinter は、指定された TM プリンターで Status API の制御をできるのに対して、

BiCloseMonPrinter は、オープン時に取得したハンドルでステータスを監視する TM プリンターを解除します。

構文

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

パラメーター

nHandle : ハンドルを指定します。

戻り値

マクロ定義 (定数)	値	説明
SUCCESS	0	正常に実行
ERR_HANDLE	-60	ハンドル値が不正
ERR_PH_NOT_EXIST	-340	PortHandler が起動していないまたは、PortHandler のクライアント、サーバー間の通信エラー




参考

戻り値による対処方法は、[17 ページ「戻り値」](#) を参照してください。

BiLockPrinter

TM プリンターを占有します。

 参考	本 API は共有プリンターの場合に使用します。 ローカルプリンターの場合は、マルチプロセスの場合に使用します。
---	---

構文


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

パラメーター

nHandle: ハンドルを指定します。
 timeout: タイムアウト時間を msec(ミリ秒) 単位で設定します。正の値で指定してください。

戻り値

マクロ定義 (定数)	値	説明
SUCCESS	0	正常に実行
ERR_HANDLE	-60	ハンドル値が不正
ERR_EXEC_FUNCTION	-310	他の API が実行中のため使用できない
ERR_PH_NOT_EXIST	-340	PortHandler が起動していない または、PortHandler のクライアント、サーバー間の通信エラー
ERR_RESET	-400	TM プリンターリセット中のため使用できない
ERR_LOCKED	-1000	TM プリンターがロックされているため実行できない

 参考	戻り値による対処方法は、 17 ページ「戻り値」 を参照してください。
---	---

説明

本 API を実行すると、TM プリンターを占有することができます。これを解除する API は、BiUnlockPrinter です。その間 TM プリンターは、他のプロセスからの排他的 (デバイスに直接アクセスする) API を受け付けません。これらの API には ERR_LOCKED が返ります。

TM プリンターの占有権はプロセスに対して与えられます。そのため占有中に同一プロセスの他のスレッドからは排他的 API を実行することができます。

占有しているプロセスからこの API を繰り返し実行できます。この場合は多重占有状態になります。占有を解除するには、本 API を実行した回数と同じ回数の BiUnlockPrinter を実行してください。

Ethernet 接続や共有プリンターをクライアントから占有する場合、占有中に TM プリンターとの接続が失われると占有が一時的に解除され、接続が復活すると自動的に再び占有状態に戻ります。

ただし接続が切れている間は、別のプロセスが TM プリンターを占有することができます。この間、元のプロセスが排他的 API を実行すると ERR_LOCKED が返ります。また別のプロセスが占有を終了すると、自動的に元のプロセスの占有状態に戻ります。

接続が失われる原因としては、以下の要因が挙げられます。

【Ethernet 接続の TM プリンターを占有する場合】


- TM プリンターの電源を OFF、コンピューターと TM プリンター間の Ethernet 接続断
- コンピューターのサスペンド / 休止

【共有プリンターをクライアントコンピューターから占有する場合】

- クライアントコンピューターとサーバーコンピューター間の接続断
- クライアントコンピューターのサスペンド / 休止

BiUnlockPrinter

TM プリンターの占有を解除します。

 参考	本 API は共有プリンターの場合に使用します。 ローカルプリンターの場合は、マルチプロセスの場合に使用します。
---	---

構文


int **BiUnlockPrinter** (int nHandle)

パラメーター

nHandle: ハンドルを指定します。

戻り値

マクロ定義 (定数)	値	説明
SUCCESS	0	正常に実行
ERR_HANDLE	-60	ハンドル値が不正
ERR_EXEC_FUNCTION	-310	他の API が実行中のため使用できない
ERR_PH_NOT_EXIST	-340	PortHandler が起動していない または、PortHandler のクライアント、サーバー間の通信エラー
ERR_RESET	-400	TM プリンターリセット中のため使用できない
ERR_LOCKED	-1000	TM プリンターがロックされているため実行できない

 参考	戻り値による対処方法は、 17 ページ「戻り値」 を参照してください。
---	---

説明

この API は、BiLockPrinter で占有した TM プリンターの占有を解除します。解除すると、TM プリンターは他のプロセスからの API を受け付けることができます。本 API を TM プリンターの占有をしていないときに実行した場合、戻り値に ERR_LOCKED が返されます。

BiDirectIOEx

TM プリンターに特定の命令 (ESC/POS コマンド) を送ります。命令の実行結果を TM プリンターから取得することもできます。また、動作中に ASB ステータスの読み込みを制御できます。本 API が終了するまで TM プリンターから ASB ステータスを送信しない制御ができるので、TM プリンターから実行結果を受け取る場合は、こちらをお勧めいたします。



参考

コマンドの詳細は、ESC/POS コマンドリファレンスを参照してください。
https://reference.epson-biz.com/modules/ref_escpos_ja/

構文

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

パラメーター

nHandle:	ハンドルを指定します。
writeLen:	TM プリンターに書き込むデータ長を指定します。"0" を指定した場合、書き込みは行いません。
writeCmd:	TM プリンターに書き込むデータ (ESC/POS コマンド) を指定します。
readLen:	TM プリンターから取得するデータの長さを指定します。TM プリンターからのコマンド実行結果が必要な場合に指定します。取得したデータのバッファーサイズが返されます。 必要のない場合は、"0" を指定します。
readBuff:	TM プリンターから取得したデータを保存するバッファーを指定します。
timeout:	タイムアウト時間を msec (ミリ秒) 単位で指定します。
nullTerminate:	"True" の場合、TM プリンターから NULL を取得した時点で取得を終了します。この時、readLen に readBuff のサイズを指定してください。 "False" の場合、readLen で指定された長さのデータを取得、またはタイムアウトエラーが発生するまで TM プリンターからデータを取得します。
option:	ASB ステータスの取得を制御します。

値	説明
0	ASB ステータスを取得しません。
1	データ取得後、ASB ステータスを取得します。



参考

- writeLen/readLen は最大 2GB まで指定可能ですが、必要最小限のデータ長を指定してください。
- readBuff のサイズは、readLen で指定した長さと同じか、それ以上のサイズを確保してください。

戻り値

マクロ定義（定数）	値	説明
SUCCESS	0	正常に実行
ERR_NO_MEMORY	-50	メモリーが足りない
ERR_HANDLE	-60	ハンドル値が不正
ERR_TIMEOUT	-70	タイムアウトエラー
ERR_ACCESS	-80	TM プリンターにアクセスできない (他アプリケーションがアクセス中もしくは、印刷中)
ERR_PARAM	-90	パラメーターエラー
ERR_BUFFER_OVER_FLOW	-140	バッファ容量不足
ERR_EXEC_FUNCTION	-310	他の API が実行中のため使用できない
ERR_PH_NOT_EXIST	-340	PortHandler が起動していない または、PortHandler のクライアント、サーバー間の通信エラー
ERR_RESET	-400	TM プリンターリセット中のため使用できない
ERR_LOCKED	-1000	TM プリンターがロックされているため実行できない



参考

戻り値による対処方法は、[17 ページ「戻り値」](#)を参照してください。

説明

本 API が正常に実行されたか確認するには、本 API の戻り値および、コマンドが正常に実行されたか TM プリンターの動作を確認してください。TM プリンターからの実行結果を取得した場合（readLen を指定）は、その実行結果を確認してください。

本 API を呼び出したときの、TM プリンターの状態によって以下の動作をします。

TM プリンターの状態	動作
オンライン	戻り値に " SUCCESS " を返します。コマンドを実行します。
オフライン	timeout 時間内で送受信が成功した場合： 戻り値に " SUCCESS " を返します。 timeout 時間内の送受信に失敗した場合： 戻り値に " ERR_TIMEOUT " を返します。
ケーブルが接続されていない / 電源オフ	戻り値に " ERR_ACCESS " を返します。
印刷中	戻り値に " ERR_LOCKED " を返します。


注意

- パラメーターの option は、TM プリンターからの応答を求めるコマンドを送る場合に、目的としないデータを受信しないために指定します。"0" を指定した場合は、目的としないデータを受信することを考慮したプログラミングをしてください。
- 受信バッファの指定有無は、TM プリンターからの受信データを本 API で処理するか、監視スレッド (BiGetStatus 関数など) で処理するのと同等の処理です。次を参照してください。

送信コマンド	受信バッファ指定	受信バッファ	監視スレッドの動作
ステータス取得コマンド	有り	ASB ステータスは受信バッファに保存される	<ul style="list-style-type: none"> コールバックされない ステータスの更新がされない
	無し	-	<ul style="list-style-type: none"> コールバックされる ステータスの更新がされる
その他 TM プリンターから応答のあるコマンド	有り	TM プリンター応答が受信バッファに入れられる	監視スレッドに影響なし
	無し	-	異常なコールバックが発生する可能性がある
その他 TM プリンターから応答のないコマンド	有り	タイムアウトエラー発生	監視スレッドに影響なし
	無し	-	監視スレッドに影響なし

BiResetPrinter

ステータスを監視している TM プリンターをリセットします。

 参考	<ul style="list-style-type: none"> 印刷中に呼び出した場合、印刷ジョブがキャンセルされます。 シリアルインターフェイスの TM プリンターは、リセットできません。
---	--

構文


int *BiResetPrinter* (int nHandle)

パラメーター

nHandle: ハンドルを指定します。


戻り値

マクロ定義 (定数)	値	説明
SUCCESS	0	正常に実行
ERR_HANDLE	-60	ハンドル値が不正
ERR_NOT_SUPPORT	-100	サポートしていない機種
ERR_EXEC_FUNCTION	-310	他の API が実行中のため使用できない
ERR_PH_NOT_EXIST	-340	PortHandler が起動していない、または、PortHandler のクライアント、サーバー間の通信エラー
ERR_RESET	-400	TM プリンターリセット中のため使用できない
ERR_LOCKED	-1000	TM プリンターがロックされているため実行できない

 参考	戻り値による対処方法は、 17 ページ「戻り値」 を参照してください。
---	---

説明

本 API が正常に実行されたか確認するには、本 API の戻り値および、TM プリンターがリセットされてオンラインになる (ASB ステータスで確認) ことを確認してください。

 参考	本 API を実行後、15 秒間は印刷を受け付けません。その間に印刷を要求した場合には、スプーラーに JOB が蓄積され、上記の時間経過後に印刷処理が行われます。
---	---

本 API を呼び出したときの、TM プリンターの状態によって以下の動作をします。

TM プリンターの状態	動作
オンライン	戻り値に " SUCCESS " を返し、リセットされます。
オフライン	戻り値に " ERR_TIMEOUT " を返し、リセットされません。
ケーブルが接続されていない / 電源オフ	ASB ステータスに " ASB_NO_RESPONSE " を返されます。リセットされません。
印刷中	印刷ジョブがキャンセルされ、リセットされます。

BiForceResetPrinter

ステータスを監視している TM プリンターを強制的にリセットします。
 マルチスレッド / マルチプロセス / マルチユーザーの環境でも、TM プリンターをリセットできます。
 別のプログラムから BiLockPrinter で占有されている場合でも、TM プリンターをリセットできます。
 ネットワークプリンターのコネクションが切れて再接続した場合、印刷できるようになるまでに時間がかかることがあります。本 API はその時間を短縮できます。



参考

- 本 API は、印刷中に呼び出した場合でも強制的に TM プリンターがリセットされ、印刷中のデータも消去されます。本 API は注意してお使いください。
- シリアルインターフェイスの TM プリンターは、リセットできません。

構文

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

パラメーター

nHandle: ハンドルを指定します。

戻り値

マクロ定義 (定数)	値	説明
SUCCESS	0	正常に実行
ERR_HANDLE	-60	ハンドル値が不正
ERR_ACCESS	-80	TM プリンターにアクセスできない
ERR_NOT_SUPPORT	-100	サポートしていない機種
ERR_EXEC_FUNCTION	-310	他の API が実行中のため使用できない
ERR_PH_NOT_EXIST	-340	PortHandler が起動していない、 または、PortHandler のクライアント、サーバー間の通信エラー




参考

戻り値による対処方法は、[17 ページ「戻り値」](#)を参照してください。

BiGetType

TM プリンターのタイプ ID を取得します。

 参考	取得できるタイプ ID については、ご購入元にお問い合わせください。
---	------------------------------------

構文


int **BiGetType** (int nHandle, LPBYTE typeId, LPBYTE font, LPBYTE exrom, LPBYTE special)

パラメーター

nHandle: ハンドルを指定します。
 typeId: TM プリンターのタイプ ID が返されます。
 font: デバイスフォントが返されます。
 exrom: 使用しません。
 special: TM プリンターの特殊 ID がセットされます。

戻り値

マクロ定義 (定数)	値	説明
SUCCESS	0	正常に実行
ERR_HANDLE	-60	ハンドル値が不正
ERR_TIMEOUT	-70	タイムアウトエラー
ERR_ACCESS	-80	TM プリンターにアクセスできない (他アプリケーションがアクセス中もしくは、印刷中)
ERR_PARAM	-90	パラメーターエラー
ERR_NOT_SUPPORT	-100	サポートしていない機種
ERR_EXEC_FUNCTION	-310	他の API が実行中のため使用できない
ERR_PH_NOT_EXIST	-340	PortHandler が起動していない または、PortHandler のクライアント、サーバー間の通信エラー
ERR_RESET	-400	TM プリンターリセット中のため使用できない
ERR_LOCKED	-1000	TM プリンターがロックされているため実行できない

 参考	戻り値による対処方法は、 17 ページ「戻り値」 を参照してください。
---	---

説明

機種によっては取得できない情報があります。その場合、typeID に 0 が返されます。

BiGetStatus

現在の TM プリンターの状態 (ASB ステータス) を取得します。

構文

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

パラメーター

nHandle: ハンドルを指定します。

lpStatus: Status API が保持している ASB ステータスが返されます。

ASB ステータスは、4 バイトで構成されています。

戻り値

マクロ定義 (定数)	値	説明
SUCCESS	0	正常に実行
ERR_HANDLE	-60	ハンドル値が不正
ERR_PARAM	-90	パラメーターエラー
ERR_EXEC_FUNCTION	-310	他の API が実行中のため使用できない



参考

戻り値による対処方法は、[17 ページ「戻り値」](#)を参照してください。

説明

各 TM プリンターの取得できる ASB ステータスは、「プリンター仕様」を参照してください。

BiSetStatusBackFunction

自動ステータス通知の際に呼び出されるコールバック関数を登録します。

構文

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

パラメーター

nHandle: ハンドルを指定します。

*pFunction: ASB ステータスを通知する、コールバック関数のアドレスをセットします。

コールバック関数のパラメーター

dwStatus: コールバック関数に Status API が保持している ASB ステータスが返されます。
ASB ステータスは、4 バイトで構成されています。

戻り値

マクロ定義 (定数)	値	説明
SUCCESS	0	正常に実行
ERR_HANDLE	-60	ハンドル値が不正
ERR_PARAM	-90	パラメーターエラー
ERR_EXIST	-210	指定されたデータがすでに存在する
ERR_EXEC_FUNCTION	-310	他の API が実行中のため使用できない



参考

戻り値による対処方法は、[17 ページ「戻り値」](#)を参照してください。

説明

本 API を呼び出すと、TM プリンターの状態を dwStatus にセットしコールバック関数が呼ばれます。TM プリンターの状態が変化すると、自動的に dwStatus に新しい情報をセットしコールバック関数が呼び出されます。本 API の解除は、BiCancelStatusBack を使用してください。

各 TM プリンターの取得できる ASB ステータスは、「プリンター仕様」を参照してください。



参考

登録したコールバック関数内から、Status API を使用することはできません。

BiSetStatusBackFunctionEx

自動ステータス通知の際に呼び出されるコールバック関数を登録します。

BiSetStatusBackFunction の機能に加えて、どのプリンターポートからのコールバックかを認識できます。

構文

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

パラメーター

nHandle: ハンドルを指定します。

*pFunction: ASB ステータスを通知する、コールバック関数のアドレスをセットします。

コールバック関数のパラメーター

dwStatus: コールバック関数に Status API が保持している ASB ステータスが返されます。
ASB ステータスは、4 バイトで構成されています。

lpcPortName: コールバックされたプリンターポート名が返されます。

戻り値

マクロ定義 (定数)	値	説明
SUCCESS	0	正常に実行
ERR_HANDLE	-60	ハンドル値が不正
ERR_PARAM	-90	パラメーターエラー
ERR_EXIST	-210	指定されたデータがすでに存在する
ERR_EXEC_FUNCTION	-310	他の API が実行中のため使用できない



参考

戻り値による対処方法は、[17 ページ「戻り値」](#)を参照してください。

説明

本 API を呼び出すと、TM プリンターの状態を dwStatus にセットしコールバック関数が呼ばれます。TM プリンターの状態が変化すると、自動的に dwStatus に新しい情報をセットしコールバック関数が呼び出されます。本 API の解除は、BiCancelStatusBack を使用してください。

各 TM プリンターの取得できる ASB ステータスは、「プリンター仕様」を参照してください。



参考

登録したコールバック関数内から、Status API を使用することはできません。

BiCancelStatusBack

BiSetStatusBackFunction / BiSetStatusBackFunctionEx で呼び出した自動ステータス通知要求処理を解除します。

構文

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

パラメーター

nHandle: ハンドルを指定します。

戻り値

マクロ定義 (定数)	値	説明
SUCCESS	0	正常に実行
ERR_HANDLE	-60	ハンドル値が不正
ERR_EXEC_FUNCTION	-310	他の API が実行中のため使用できない




参考

- 戻り値による対処方法は、[17 ページ「戻り値」](#)を参照してください。
- 自動ステータス通知要求処理の登録が行われていない場合に、間違って本 API を呼び出しても戻り値は“SUCCESS”を返します。

BiPowerOff

TM プリンターを電源オフ、もしくは待機状態にします。

 参考	本 API は、TM プリンターがオフライン復帰待ち状態の場合、使用できません。また、本 API が実行中に TM プリンターがオフライン状態になった場合も実行されません。
---	--

構文


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

パラメーター

nHandle: ハンドルを指定します。

戻り値

マクロ定義 (定数)	値	説明
SUCCESS	0	正常に実行
ERR_NO_MEMORY	-50	メモリーが足りない
ERR_HANDLE	-60	ハンドル値が不正
ERR_TIMEOUT	-70	タイムアウトエラー
ERR_ACCESS	-80	TM プリンターにアクセスできない (他アプリケーションがアクセス中もしくは、印刷中)
ERR_NOT_SUPPORT	-100	サポートしていない機種
ERR_EXEC_FUNCTION	-310	他の API が実行中のため使用できない
ERR_PH_NOT_EXIST	-340	PortHandler が起動していない または、PortHandler のクライアント、サーバー間の通信エラー
ERR_LOCKED	-1000	TM プリンターがロックされているため実行できない

 参考	戻り値による対処方法は、 17 ページ「戻り値」 を参照してください。
---	---

BiGetPrnCapability

TM プリンターのプリンター ID を取得します。



参考

prnID、取得できる TM プリンター情報については、ご購入元にお問い合わせください。
サポートしていない prnID を指定した場合、タイムアウトエラーが発生します。

構文

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

パラメーター

nHandle: ハンドルを指定します。
prnID: 取得したい TM プリンター情報の prnID を指定します。
pBuffSize: TM プリンター情報をセットするメモリのサイズ (1 ~ 80) を指定します。本 API を呼び出し後は、実際に読み取ったデータのサイズを返します。バッファ容量不足の場合、必要なバイト数が返されます。
pBuff: TM プリンター情報をセットするメモリアドレスを指定します。また、指定した prnID の情報を返します。

戻り値

マクロ定義 (定数)	値	説明
SUCCESS	0	正常に実行
ERR_HANDLE	-60	ハンドル値が不正
ERR_TIMEOUT	-70	タイムアウトエラー
ERR_ACCESS	-80	TM プリンターにアクセスできない (他アプリケーションがアクセス中もしくは、印刷中)
ERR_PARAM	-90	パラメーターエラー
ERR_BUFFER_OVER_FLOW	-140	バッファ容量不足
ERR_EXEC_FUNCTION	-310	他の API が実行中のため使用できない
ERR_PH_NOT_EXIST	-340	PortHandler が起動していない または、PortHandler のクライアント、サーバー間の通信エラー
ERR_RESET	-400	TM プリンターリセット中のため使用できない
ERR_LOCKED	-1000	TM プリンターがロックされているため実行できない




参考

戻り値による対処方法は、[17 ページ「戻り値」](#)を参照してください。

BiOpenDrawer

ドロアーをオープンします。

 参考	TM プリンターがオフラインの場合でもドロアーはオープンします。
---	----------------------------------

構文

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

パラメーター

nHandle: ハンドルを指定します。

drawer: オープンするドロアーを指定します。


マクロ定義 (定数)	値	説明
EPS_BI_DRAWER_1	1	ドロアー 1 をオープン
EPS_BI_DRAWER_2	2	ドロアー 2 をオープン

pulse: ドロアーキック信号のオン時間を指定します。

マクロ定義 (定数)	値	説明
EPS_BI_PLUSE_100	1	100 ミリ秒の信号
EPS_BI_PLUSE_200	2	200 ミリ秒の信号
EPS_BI_PLUSE_300	3	300 ミリ秒の信号
EPS_BI_PLUSE_400	4	400 ミリ秒の信号
EPS_BI_PLUSE_500	5	500 ミリ秒の信号
EPS_BI_PLUSE_600	6	600 ミリ秒の信号
EPS_BI_PLUSE_700	7	700 ミリ秒の信号
EPS_BI_PLUSE_800	8	800 ミリ秒の信号

戻り値

マクロ定義 (定数)	値	説明
SUCCESS	0	正常に実行
ERR_HANDLE	-60	ハンドル値が不正
ERR_ACCESS	-80	TM プリンターにアクセスできない (他アプリケーションがアクセス中もしくは、印刷中)
ERR_PARAM	-90	パラメーターエラー
ERR_NOT_SUPPORT	-100	サポートしていない機種
ERR_EXEC_FUNCTION	-310	他の API が実行中のため使用できない
ERR_PH_NOT_EXIST	-340	PortHandler が起動していない または、PortHandler のクライアント、サーバー間の通信エラー
ERR_RESET	-400	TM プリンターリセット中のため使用できない
ERR_LOCKED	-1000	TM プリンターがロックされているため実行できない

 参考	戻り値による対処方法は、 17 ページ「戻り値」 を参照してください。
---	---

説明

本 API を呼び出したときの、TM プリンターの状態によって以下の動作をします。

TM プリンターの状態	動作
オンライン	戻り値に " SUCCESS" を返し、ドロアーオープンされます。
オフライン	戻り値に " SUCCESS" を返し、ドロアーオープンされます。
ケーブルが接続されていない / 電源オフ	戻り値に "ERR_ACCESS" を返します。ドロアーオープンされません。

.NET API リファレンス

本章では、.NET 環境で使用する Status API と構文について説明しています。

プロパティ

IsValid

TM プリンターのオープン状態を取得します。

アクセス : 読み取り専用

データ型 : System.Boolean

説明

以下の値を返します。

true : オープンの状態

false : オープンしていない状態

LastError

最後に実行した API のエラーコードを取得します。

アクセス : 読み取り専用

データ型 : com.epson.pos.driver.ErrorCode

説明

本プロパティは最後に実行した API の戻り値を保持しており、いつでも取得することができます。

プロパティはエラーコードを返せないため、本 API でエラーコードを取得します。

エラーコードの詳細は [17 ページの「戻り値」](#) を参照してください。

Status

TM プリンターの状態 (ASB ステータス) を取得します。詳細は [30 ページの「BiGetStatus」](#) を参照してください。

アクセス : 読み取り専用

データ型 : `com.epson.pos.driver.ASB`

説明

取得した ASB ステータスは、`com.epson.pos.driver.ASB` で定義されている定数で確認できます。
取得できる値は TM プリンターによって異なります。詳細は「プリンター仕様」を参照してください。

メソッド

OpenMonPrinter

指定された TM プリンターの制御を開始します。
詳細は [18 ページの「BiOpenMonPrinter」](#) を参照してください。

構文

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

パラメーター

OpenType type: name に指定する名称のタイプです。com.epson.pos.driver.OpenType で定義されている定数を指定します。
String name: プリンタードライバー名または TM プリンターの接続されているポート名を指定します。

CloseMonPrinter

Status API による TM プリンターの制御を終了します。
詳細は [20 ページの「BiCloseMonPrinter」](#) を参照してください。

構文

ErrorCode **CloseMonPrinter** ()

LockPrinter

TM プリンターを占有します。
詳細は [21 ページの「BiLockPrinter」](#) を参照してください。

構文

ErrorCode **LockPrinter** (int timeout)

パラメーター

int timeout: タイムアウト時間を msec(ミリ秒) 単位で設定します。正の値で指定してください。

UnlockPrinter

TM プリンターの占有を解除します。
 詳細は [23 ページの「BiUnlockPrinter」](#) を参照してください。

構文

ErrorCode **UnlockPrinter** ()

DirectIOEx

TM プリンターに特定の命令 (ESC/POS コマンド) を送ります。また、送った命令の実行結果を TM プリンターから取得します。
 詳細は [24 ページの「BiDirectIOEx」](#) を参照してください。

構文

- ErrorCode **DirectIOEx** (byte[] writeCmd, ref byte[] readBuff, int timeout, bool nullTerminate, byte option)
 説明: TM プリンターに特定の命令を送り、送った命令の実行結果 (バイナリーデータ) を TM プリンターから取得します。
- ErrorCode **DirectIOEx** (byte[] writeCmd, out String response, int timeout, byte option)
 説明: TM プリンターに特定の命令を送り、送った命令の実行結果 (文字列データ) を TM プリンターから取得します。
- ErrorCode **DirectIOEx** (byte[] writeCmd, int timeout)
 説明: TM プリンターに特定の命令を送るのみで、送った命令の実行結果を TM プリンターから取得しません。また、ASB ステータスも取得しません。

パラメーター

byte[] writeCmd:	TM プリンターに送信するデータ (ESC/POS コマンド) を指定します。
ref byte[] readBuff:	TM プリンターから受信するデータを保存するバッファを指定します。 また、TM プリンターからデータを受信します。
int timeout:	データ送受信のタイムアウト時間 (ミリ秒単位) を指定します。
bool nullTerminate:	"True" の場合、TM プリンターから NULL を受信した時点で読み込みを終了します。 "False" の場合、readBuff で指定されたバッファの長さ分のデータを読み込むか、タイムアウトエラーが発生するまで TM プリンターからデータを取得します。
byte option:	ASB ステータスの読み込みを指定します。
out String response:	TM プリンターから受信したデータ (文字列) です。

ResetPrinter

ステータスを監視している TM プリンターをリセットします。
詳細は [27 ページの「BiResetPrinter」](#) を参照してください。

構文

ErrorCode **ResetPrinter** ()

ForceResetPrinter

ステータスを監視している TM プリンターを強制的にリセットします。
LockPrinter で占有されている TM プリンターをリセットできます。印刷中の TM プリンターもリセットされるため、本 API は注意してお使いください。
詳細は [28 ページの「BiForceResetPrinter」](#) を参照してください。

構文

ErrorCode **ForceResetPrinter** ()

GetType

TM プリンターのタイプ ID を取得します。
詳細は [29 ページの「BiGetType」](#) を参照してください。

構文

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

パラメーター

out byte typeid: TM プリンターのタイプ ID が返されます。
out byte font: デバイスフォントが返されます。
out byte exrom: TM プリンターの拡張 Flash ROM 容量が返されます。
out byte euspecial: TM プリンターの特殊 ID がセットされます。

SetStatusBack

StatusCallback/StatusCallbackEx イベントによるステータス通知を開始します。
詳細は [32 ページの「BiSetStatusBackFunctionEx」](#) を参照してください。

構文

ErrorCode **SetStatusBack** ()

CancelStatusBack

StatusCallback/StatusCallbackEx イベントによるステータス通知を停止します。
詳細は [33 ページの「BiCancelStatusBack」](#) を参照してください。

構文

ErrorCode **CancelStatusBack** ()

PowerOff

TM プリンターを電源オフ、もしくは待機状態にします。
詳細は [34 ページの「BiPowerOff」](#) を参照してください。

構文

ErrorCode **PowerOff** ()

GetPrnCapability

TM プリンターのプリンター ID を取得します。

詳細は [35 ページ](#)の「[BiGetPrnCapability](#)」を参照してください。

構文

- **ErrorCode *GetPrnCapability*** (byte printerID, out byte[] data)
説明: プリンター ID で指定された TM プリンターの情報 (バイナリーデータ) を取得します。
- **ErrorCode *GetPrnCapability*** (byte printerID, out String data)
説明: プリンター ID で指定された TM プリンターの情報 (文字列データ) を取得します。

パラメーター

byte printerID: 取得したい TM プリンター情報 (プリンター ID) を指定します。

out byte[] data: TM プリンター情報が返されます。

out String data: TM プリンター情報が返されます。

取得する TM プリンター情報 (プリンター ID) によってデータ型が異なります。

OpenDrawer

ドロアーを制御します。

詳細は [36 ページ](#)の「[BiOpenDrawer](#)」を参照してください。

構文

ErrorCode *OpenDrawer* (Drawer drawer, Pulse pulse)

パラメーター

Drawer drawer: オープンするドロアーを指定します。

com.epson.pos.driver.Drawer で定義されている定数を指定します。

Pulse pulse: ドロアーキック信号のオン時間を指定します。

com.epson.pos.driver.Pulse で定義されている定数を指定します。

イベント

StatusCallback

通知された ASB ステータスを処理するイベントです。
詳細は [31 ページの「BiSetStatusBackFunction」](#) を参照してください。

構文

StatusCallbackHandler (ASB asb)

パラメーター

ASB asb: 通知される ASB ステータスです。
通知された ASB ステータスは、com.epson.pos.driver.ASB で定義されている定数で確認できます。詳細は、「プリンター仕様」を参照してください。

StatusCallbackEx

通知された ASB ステータスを処理するイベントです。
詳細は [32 ページの「BiSetStatusBackFunctionEx」](#) を参照してください。

構文

StatusCallbackHandlerEx (ASB asb, String portName)

パラメーター

ASB asb: 通知される ASB ステータスです。
通知された ASB ステータスは、com.epson.pos.driver.ASB で定義されている定数で確認できます。詳細は、「プリンター仕様」を参照してください。
String portName: コールバックされたポート名が通知されます。

付録

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

Microsoft 社の Windows Template Library を使用しています。

ご注意

- (1) 本書の内容の一部または全部を無断で転載、複写、複製、改ざんすることは固くお断りします。
- (2) 本書の内容については、予告なしに変更することがあります。
- (3) 本書の内容については、万全を期して作成いたしましたが、万一ご不審な点や誤り、記載もれなど、お気づきの点がありましたらご連絡ください。
- (4) 運用した結果の影響については、上項に関わらず責任を負いかねますのでご了承ください。
- (5) 本製品がお客様により不適切に使用されたり、本書の内容に従わずに取り扱われたり、またはエプソンおよびエプソン指定の者以外の第三者により修理・変更されたことなどに起因して生じた損害などにつきましては、責任を負いかねますのでご了承ください。
- (6) エプソン純正品およびエプソン品質認定品以外のオプションまたは消耗品を装着してトラブルが発生した場合には、責任を負いかねますのでご了承ください。

商標

EPSON、EXCEED YOUR VISION、および ESC/POS はセイコーエプソン株式会社の登録商標です。
Microsoft[®]、Windows[®]、Visual Studio[®]、Visual C++[®]、Visual C#[®] は米国 Microsoft Corporation の米国およびその他の国における商標または登録商標です。
その他の製品名および会社名は、各社の商標または登録商標です。

ESC/POS[®] コマンドシステム

EPSON は、独自の POS プリンターコマンドシステム、ESC/POS により、業界のイニシアチブをとってきました。ESC/POS は特許取得済みのものを含む数多くの独自のコマンドを持ち、高い拡張性で多才な POS システムの構築を実現します。ほとんどの EPSON POS プリンターとディスプレイに互換性を持つほか、この独自の制御システムにはフレキシビリティもあるため、将来アップグレードが行いやすくなります。その機能と利便性は世界中で評価されています。

© Seiko Epson Corporation 2019 - 2021.