

CITIZEN

Windows Label Print SDK

プログラムマニュアル

Ver. 1.05 用

シチズン・システムズ株式会社

更新履歴

年月日	バージョン	履歴
2016/04/26	1.00	新規
2017/03/03	1.01	<ul style="list-style-type: none"> ・CL-E720/730 モデルのインターフェースに無線 LAN を追加 ・名前空間を「com.citizen.sdk.LabelPrint」に変更 ・Connect, SearchCitizenPrinter, SearchESCPOSPrinter メソッドの接続タイプに Bluetooth を追加 ・PrinterCheck メソッドに ParallelErrorOutput の説明を追加 ・Preview メソッドを追加 ・DrawTextPCFont メソッドの引数に measurementUnit を追加 ・DrawBitmap メソッドの引数に resolution, measurementUnit を追加
2017/09/29		<ul style="list-style-type: none"> ・新モデル CL-E300/CL-E303 を追加 ・Windows ドライバー名を更新
2018/01/24		<ul style="list-style-type: none"> ・新モデル CL-E321/CL-E331 を追加 ・Windows ドライバー名を更新
2018/09/18	1.02	<ul style="list-style-type: none"> ・CL-E720/730 モデルのインターフェースに Bluetooth を追加 ・DrawTextPtrFont に中国モデル、韓国モデル用のロケールを追加 ・ロケールの定義に中国モデル、韓国モデルを追加
2019/08/28	1.03	<ul style="list-style-type: none"> ・PrinterCheck メソッドの説明および使用例を修正 ・LabelPrinter クラスに PrintHeadLowTemp、PrintHeadFailure、PrintHeadOverheat、MechanismOpen、AutoCutterError、FanMotorError、MiscError プロパティを追加 ・DrawTextPCFont メソッドの resolution、measurementUnit パラメータの説明を修正 ・DrawBitmap メソッドの measurementUnit パラメータの説明を修正
2019/11/12		<ul style="list-style-type: none"> ・新モデル CL-S700II/703II、CL-S620II/630II、CL-S520II/530II を追加 ・Windows ドライバーバージョンの記述を、ドライバ名 の記述に変更 ・Windows ドライバー対象モデルのモデル名を修正 ・Windows ドライバーステータス機能の無効化について修正
2020/06/02		<ul style="list-style-type: none"> ・新モデル CL-E300EX/303EX、CL-E321EX/331EX を追加 ・Windows ドライバーバージョンの記述を、ドライバ名 の記述に変更
2021/11/26	1.04	<ul style="list-style-type: none"> ・対応オペレーティングシステムに Windows11 を追加
2023/12/22		<ul style="list-style-type: none"> ・新モデル CL-S700III/703III を追加
2024/04/12	1.05	<ul style="list-style-type: none"> ・StoreNVBitmap および DrawBitmap メソッドにビットマップモード指定を追加 (27,80 頁) ・PrinterCheckEx および ClearOutputEx メソッドを追加 (34,37 頁) ・DrawTextPtrFont および DrawTextDLFont メソッドの UTF-8 対応 (73,75 頁)

ご注意

1. 本書の内容の一部、または全部を無断で転載することは、固くお断りいたします。
2. 本書の内容については、事前の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、万一誤り・お気付きの点がございましたら、ご連絡くださいますようお願いいたします。
4. 運用した結果の影響につきましては、3項にかかわらず責任を負いかねますのでご了承ください。
5. 上記に同意いただけない場合は、本SDKをご使用いただけません。

商標

Microsoft、Windows、Visual Studio、Visual Basic、Visual C#、Visual C++は、米国 Microsoft Corporation(あるいは米国マイクロソフト・コーポレーション)の米国およびその他の国における登録商標です。(Windows の正式名称は Microsoft Windows Operating System です)

その他、記載されている会社名、製品名は、各社の商標または登録商標です。

CITIZEN は、シチズン時計株式会社の登録商標です。

目次

1. はじめに	6
1.1. ドキュメント対象範囲	6
1.2. 動作環境	6
1.3. 定義方法	9
1.4. プログラム構造	11
1.5. 機能一覧	13
2. ライブラリインターフェイス	17
2.1. 戻り値	17
2.2. LabelPrinter クラス	18
2.2.1. コンストラクタ	18
2.2.2. Connect メソッド	19
2.2.3. Disconnect メソッド	21
2.2.4. SetCommProperties メソッド	22
2.2.5. PrinterCheck メソッド	23
2.2.6. Print メソッド	25
2.2.7. Preview メソッド	26
2.2.8. StoreNVBitmap メソッド	27
2.2.9. ClearOutput メソッド	29
2.2.10. SendData メソッド	30
2.2.11. SearchCitizenPrinter メソッド	31
2.2.12. SearchLabelPrinter メソッド	33
2.2.13. PrinterCheckEx メソッド	34
2.2.14. ClearOutputEx メソッド	37
2.2.15. SetLog メソッド	38
2.2.16. HorizontalMagnification プロパティ	39
2.2.17. VerticalMagnification プロパティ	40
2.2.18. FormatAttribute プロパティ	41
2.2.19. ContinuousMediaLength プロパティ	42
2.2.20. MeasurementUnit プロパティ	43
2.2.21. PrintSpeed プロパティ	44
2.2.22. FeedSpeed プロパティ	45
2.2.23. SlewSpeed プロパティ	46
2.2.24. BackupSpeed プロパティ	47
2.2.25. PrintDarkness プロパティ	48
2.2.26. DoubleHeat プロパティ	49
2.2.27. VerticalOffset プロパティ	50
2.2.28. HorizontalOffset プロパティ	51
2.2.29. MediaHandling プロパティ	52
2.2.30. StartOffset プロパティ	53
2.2.31. StopOffset プロパティ	54
2.2.32. LabelSensor プロパティ	55
2.2.33. PrintMethod プロパティ	56
2.2.34. SensorLocation プロパティ	57
2.2.35. CommandInterpreterInAction プロパティ	58
2.2.36. PaperError プロパティ	59
2.2.37. RibbonEnd プロパティ	60
2.2.38. BatchProcessing プロパティ	61
2.2.39. Printing プロパティ	62

2.2.40. Pause プロパティ.....	63
2.2.41. WaitingForPeeling プロパティ.....	64
2.2.42. PrintHeadLowTemp プロパティ	65
2.2.43. PrintHeadFailure プロパティ.....	66
2.2.44. PrintHeadOverheat プロパティ	67
2.2.45. MechanismOpen プロパティ	68
2.2.46. AutoCutterError プロパティ	69
2.2.47. FanMotorError プロパティ.....	70
2.2.48. MiscError プロパティ.....	71
2.3. LabelDesign クラス	72
2.3.1. コンストラクタ	72
2.3.2. DrawTextPtrFont メソッド	73
2.3.3. DrawTextDLFont メソッド.....	75
2.3.4. DrawTextPCFont メソッド.....	77
2.3.5. DrawNVBitmap メソッド	79
2.3.6. DrawBitmap メソッド.....	80
2.3.7. DrawBarCode メソッド.....	82
2.3.8. DrawMaxiCode メソッド	86
2.3.9. DrawPDF417 メソッド.....	87
2.3.10. DrawDataMatrix メソッド	88
2.3.11. DrawQRCode メソッド.....	89
2.3.12. DrawAztec メソッド.....	90
2.3.13. DrawGS1DataBar メソッド.....	91
2.3.14. DrawLine メソッド.....	93
2.3.15. DrawRect メソッド.....	94
2.3.16. FillRect メソッド.....	95
2.3.17. DrawCircle メソッド.....	96
2.3.18. FillCircle メソッド.....	97
2.3.19. DrawPolygon メソッド.....	98
2.3.20. FillPolygon メソッド.....	99
2.3.21. EmbedRawDesignCommand メソッド.....	100
3. 補足	101
3.1. 印字位置指定	101
3.2. ログ機能について	102
3.3. パラメータ	104

1. はじめに

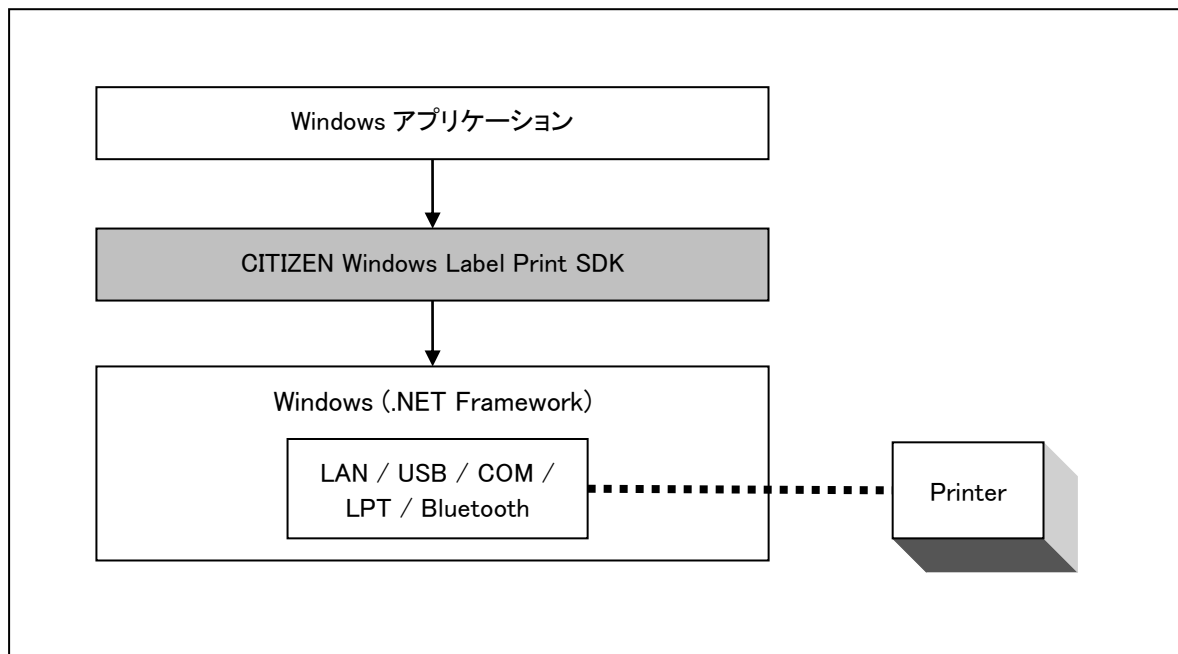
本ドキュメントは、CITIZEN Windows Label Print SDK のプログラムマニュアルです。

1.1. ドキュメント対象範囲

本ドキュメントは、CITIZEN Label プリンターを利用する Windows アプリケーションの開発者が参照することを目的としています。

1.2. 動作環境

本 SDK は CITIZEN Label プリンターを利用する Windows アプリケーションから参照されることを想定しています。



SDK システム構成図

SDK 構成

本 SDK は、以下の3つのファイルで構成されています。ファイルをアプリケーションと同じフォルダに格納してご使用ください。

- ・ CSJLabelLib.dll : 参照定義用拡張ライブラリ
- ・ CSJLabelLibW32.dll : 32bit 用拡張ライブラリ (CSJLabelLib.dll から参照)
- ・ CSJLabelLibW64.dll : 64bit 用拡張ライブラリ (CSJLabelLib.dll から参照)

対象オペレーティングシステム

本 SDK は、以下の Microsoft Windows オペレーティングシステムに対応しています。

- ・ Windows XP
- ・ Windows 7 (32bit, 64bit)
- ・ Windows 8 (32bit, 64bit)
- ・ Windows 8.1 (32bit, 64bit)
- ・ Windows 10 (32bit, 64bit)
- ・ Windows 11 (64bit)

対象 .NET Framework

本 SDK は、以下の Microsoft .NET Framework に対応しています。

- ・ Microsoft .NET Framework 2.0 SP2

- Microsoft .NET Framework 3.5 SP1
- Microsoft .NET Framework 4.0
- Microsoft .NET Framework 4.5
- Microsoft .NET Framework 4.6
- Microsoft .NET Framework 4.7
- Microsoft .NET Framework 4.8

対象モデル

本 SDK の対象モデルおよびそのモデルに対応するインターフェースは以下の通りです。
各モデルの機能詳細についてはプリンターの取扱説明書をご参照ください。

対象モデル	インターフェース	プリンター機能
CL-S700/703	有線/無線 LAN, USB, COM, LPT	感熱/熱転写
CL-S700II/703II	有線/無線 LAN, USB, COM, LPT	感熱/熱転写
CL-S700III/703III	有線/無線 LAN, USB, COM, LPT	感熱/熱転写
CL-E720/730	有線/無線 LAN, USB, COM, LPT, Bluetooth	感熱/熱転写
CL-S620/630	有線/無線 LAN, USB, COM, LPT	感熱/熱転写
CL-S620II/630II	有線/無線 LAN, USB, COM, LPT	感熱/熱転写
CL-S520/530	有線/無線 LAN, USB, COM, LPT	感熱
CL-S520II/530II	有線/無線 LAN, USB, COM, LPT	感熱
CL-S400DT	有線/無線 LAN, USB, COM, LPT, Bluetooth	感熱
CL-E300/CL-E303	有線 LAN, USB, COM	感熱
CL-E300EX/CL-E303EX	有線/無線 LAN, USB, COM, Bluetooth	感熱
CL-E321/CL-E331	有線 LAN, USB, COM	感熱/熱転写
CL-E321EX/CL-E331EX	有線/無線 LAN, USB, COM, Bluetooth	感熱/熱転写

USB インターフェース接続について

本 SDK を USB インターフェース接続にてご使用になる際は、プリンタードライバーのインストールを行う必要があります。

パラレルインターフェース接続について

端末とプリンターがパラレルインターフェースで接続されていると、毎回端末を立ち上げた際にプリンタードライバーのインストールを要求するインストールウィザードが起動することがあります。このインストールウィザードを起動させないためには、Windows ドライバーのインストールを行ってください。

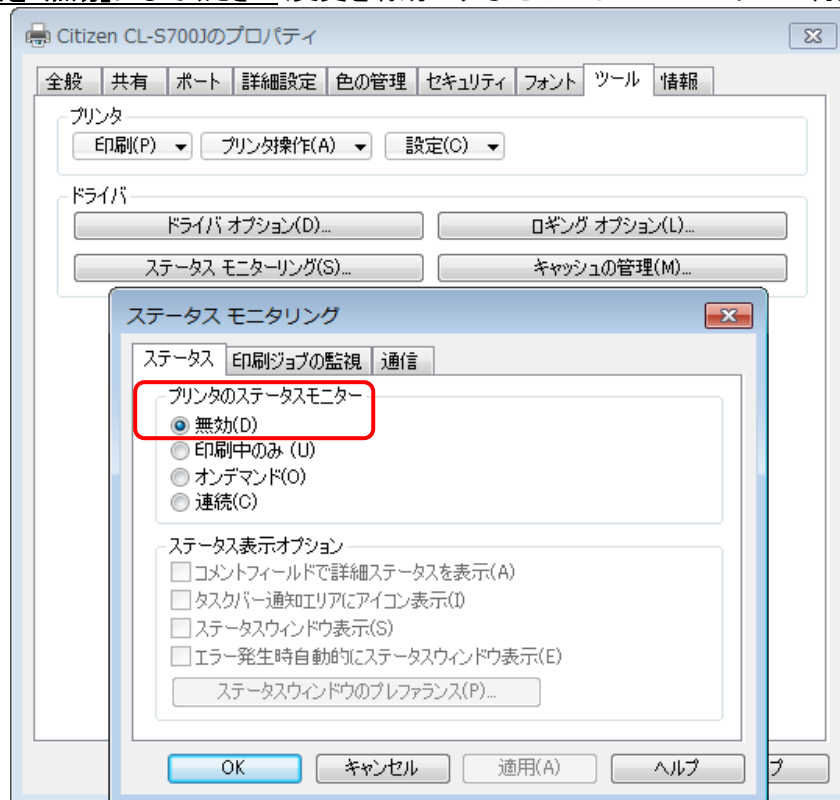
Windows ドライバーのインストール

各モデルに対応するWindowsドライバーは以下の通りです。Windowsドライバーのインストール方法については、Windowsドライバーに付属の「Windowsドライバーガイド」をご参照ください。

対象モデル	Windows ドライバー名
CL-S700/703	Citizen DPL Driver
CL-S700II/703II	
CL-S700III/703III	
CL-E720/730	
CL-S620/630	
CL-S620II/630II	

CL-S520/530	
CL-S520II/530II	
CL-S400DT	
CL-E300/303	
CL-E300EX/303EX	
CL-E321/331	
CL-E321EX/331EX	

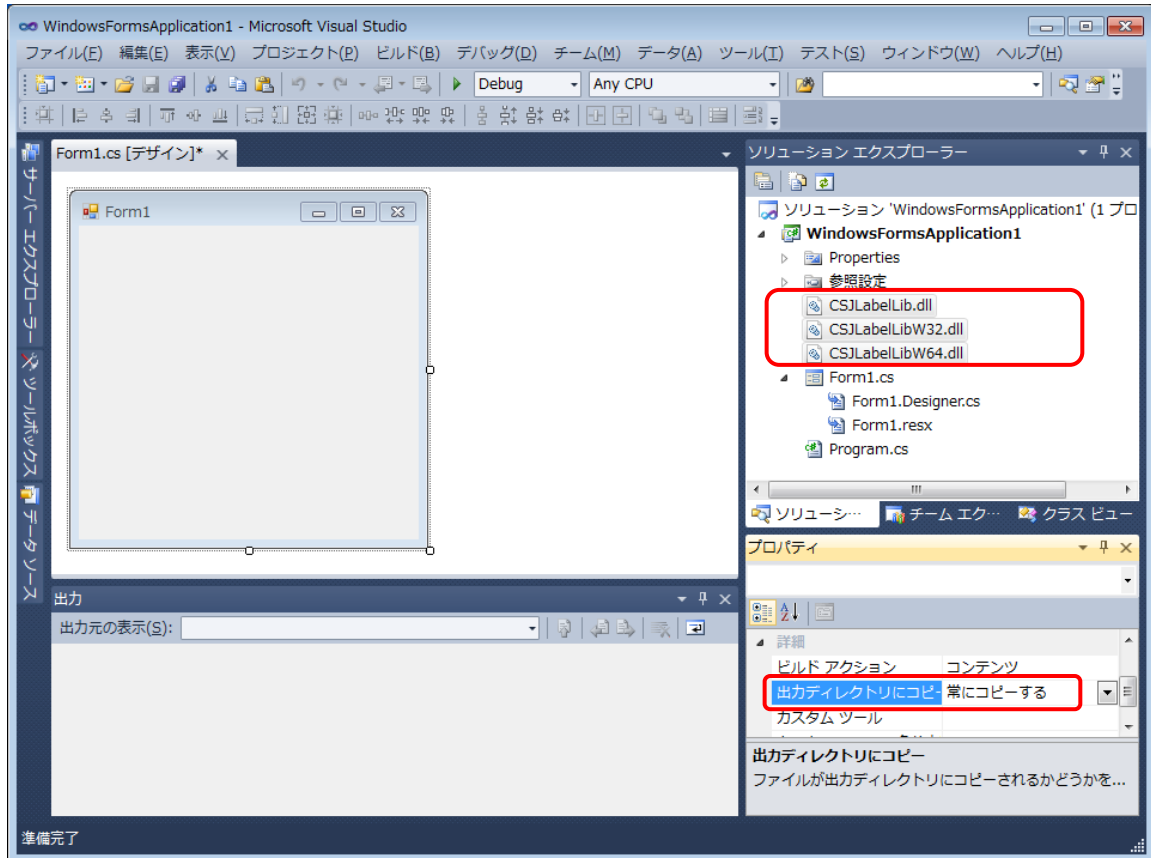
※ 本ドライバーは、Windows ドライバーのプリンターの状態を確認できるステータス機能（ステータスマニター）との併用はできません。本ドライバーをご使用の際は、**Windows ドライバーのプリンターのステータスマニター設定を「無効」にしてください**（変更を有効にするためにはコンピュータの再起動が必要です）。



1.3. 定義方法

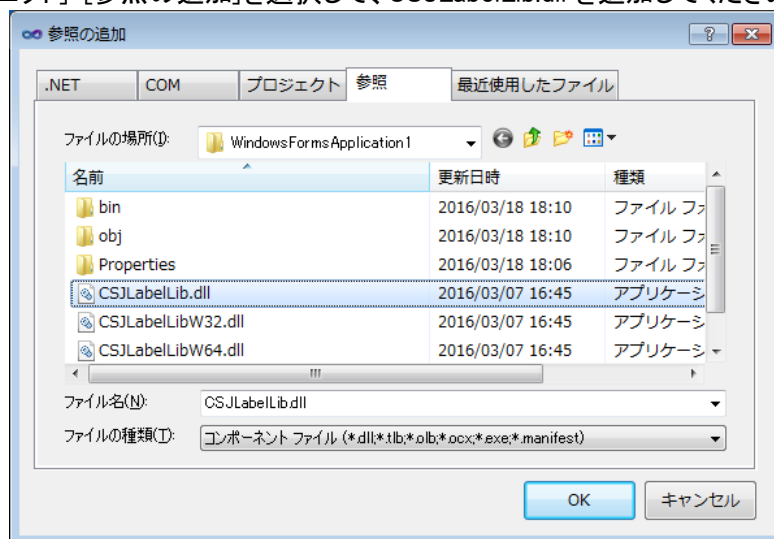
ライブラリの追加

開発するプロジェクトにライブラリファイルを追加します。VisualStudio の[ソリューション エクスプローラー]ウィンドウのプロジェクトに3つのライブラリファイルをドラッグします。(次の画面は VisualStudio2010 の C#プロジェクトの例です。)追加した3つのライブラリファイルについて、[プロパティ]ウィンドウの[出力ディレクトリにコピー]の設定を[常にコピーする]に選択してください。



参照の追加

メニューの[プロジェクト]-[参照の追加]を選択して、CSJLabelLib.dll を追加してください。



名前空間の定義

本 SDK を使用する場合は、プログラムの先頭で名前空間「com.citizen.sdk.LabelPrint」を定義してください。

C#の場合:

```
using com.citizen.sdk.LabelPrint;
```

Basic の場合:

```
Imports com.citizen.sdk.LabelPrint
```

C++の場合:

```
using namespace com::citizen::sdk::LabelPrint;
```

1.4. プログラム構造

本 SDK を使用する場合のプログラム構造は、以下の通りです。

```
// LabelDesignクラス インスタンス生成
LabelDesign design = new LabelDesign();

// テキスト描画
design.DrawTextPtrFont("Sample Print",
    LabelConst.CLS_LOCALE_JP, LabelConst.CLS_PRT_FNT_TRIUMVIRATE_B,
    LabelConst.CLS_RT_NORMAL, 1, 1, LabelConst.CLS_PRT_FNT_SIZE_24, 20,
    300);

// QRコード描画
design.DrawQRCode("DrawQRCode",
    LabelConst.CLS_ENC_CDPG_IBM850, LabelConst.CLS_RT_NORMAL, 4,
    LabelConst.CLS_QRCODE_EC_LEVEL_H, 20, 220);

// 矩形描画
design.FillRect(20, 150, 350, 40, LabelConst.CLS_SHADED_PTN_11);

// バーコード描画
design.DrawBarCode("0123456789",
    LabelConst.CLS_BCS_CODE128, LabelConst.CLS_RT_NORMAL,
    3, 3, 30, 20, 70, LabelConst.CLS_BCS_TEXT_SHOW);

// LabelPrinterクラス インスタンス生成
LabelPrinter printer = new LabelPrinter();

// シリアルポート設定(シリアル接続のみ)
printer.SetCommProperties(LabelConst.CLS_COM_BAUDRATE_9600,
    LabelConst.CLS_COM_PARITY_NONE, LabelConst.CLS_COM_HANDSHAKE_DTRDSR);

// 接続処理
int result = printer.Connect(LabelConst.CLS_PORT_COM, "COM1:");
if (LabelConst.CLS_SUCCESS == result)
{
    // プロパティ取得
    int printDarkness = printer.GetPrintDarkness();
    if (LabelConst.CLS_PROPERTY_DEFAULT == printDarkness)
    {
        // プロパティ設定
        printer.SetPrintDarkness(10);
    }

    // 印刷処理
    result = printer.Print(design, 0001);

    // 切断処理
    printer.Disconnect();

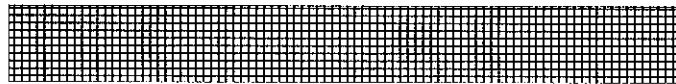
    if (LabelConst.CLS_SUCCESS == result)
    {
        // 印刷処理エラー
        MessageBox.Show("Print Error : " + result.ToString(),
            "Citizen_Label_sample", MessageBoxButtons.OK,
            MessageBoxIcon.Error);
    }
}
else
{
    // 接続処理エラー
    MessageBox.Show("Connect Error : " + result.ToString(),
        "Citizen_Label_sample", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
}
```

ラベルの
デザイン

ラベルの印刷
(接続、印刷、
切断)

印刷結果

Sample Print



0123456789

1.5. 機能一覧

本 SDK は以下の機能を提供します。

LabelPrinter クラス メソッド一覧

No	機能	詳細
1	クラス生成 (コンストラクタ)	コンストラクタです。
2	プリンター接続処理 (Connect メソッド)	プリンターと接続します。
3	プリンター切断処理 (Disconnect メソッド)	プリンターとの接続を切断します。
4	シリアル接続条件設定処理 (SetCommProperties メソッド)	シリアル接続の条件を設定します。
5	プリンター状態確認処理 (PrinterCheck メソッド)	プリンターのステータスを取得します。
6	印刷処理 (Print メソッド)	デザインしたラベルを印刷します。
7	プレビュー処理 (Preview メソッド)	デザインしたラベルをプレビューします。
8	NV ビットマップ登録処理 (StoreNVBitmap メソッド)	ビットマップ画像をフラッシュメモリへ登録します。
9	プリントバッファクリア処理 (ClearOutput メソッド)	プリンターをリセットしてプリントバッファをクリアします。
10	バイトデータ送信処理 (SendData メソッド)	バイトデータをプリンターへ送信します。
11	プリンター検索 (SearchCitizenPrinter メソッド)	プリンターを検索しプリンター情報のリストを取得します。
12	プリンター検索2 (SearchLabelPrinter メソッド)	プリンターを検索しアドレスのリストを取得します。
13	接続およびプリンター状態確認処理 (PrinterCheckEx メソッド)	接続してプリンターのステータスを取得します。
14	接続およびプリントバッファクリア処理 (ClearOutputEx メソッド)	接続してプリンターをリセットしてプリントバッファをクリアします。
15	ログ設定 (SetLog メソッド)	ログ機能を設定します。

LabelPrinter クラス プロパティ一覧

No	機能	属性	詳細
1	水平ピクセルサイズの設定/取得 (HorizontalMagnification プロパティ)	R/W	水平方向のピクセルサイズを示します。
2	垂直ピクセルサイズの設定/取得 (VerticalMagnification プロパティ)	R/W	垂直方向のピクセルサイズを示します。
3	展開方法の指定設定/取得 (FormatAttribute プロパティ)	R/W	描画が重なった部分の重ね書き/白抜きを示します。
4	連続用紙長の設定/取得 (ContinuousMediaLength プロパティ)	R/W	連続紙を使用した場合のラベル長を示します。
5	単位の設定/取得 (MeasurementUnit プロパティ)	R/W	距離指定パラメータの単位を示します。
6	速度(印刷部分)の設定/取得 (PrintSpeed プロパティ)	R/W	印刷速度を示します。
7	速度(非印刷部分)の設定/取得 (FeedSpeed プロパティ)	R/W	非印刷部分の紙送り速度を示します。

8	速度(紙送り部分)の設定/取得 (SlewSpeed プロパティ)	R/W	紙送り部分の紙送り速度を示します。
9	速度(バックフィード)の設定/取得 (BackupSpeed プロパティ)	R/W	バックフィード時の速度を示します。
10	印字濃度の設定/取得 (PrintDarkness プロパティ)	R/W	印字濃度を示します。
11	ダブルヒートの設定/取得 (DoubleHeat プロパティ)	R/W	ダブルヒート(同位置に 2 回熱をかけ、より濃く印字)を示します。
12	Y 座標オフセットの設定/取得 (VerticalOffset プロパティ)	R/W	印字開始位置における、Y 座標のオフセットを示します。
13	X 座標オフセットの設定/取得 (HorizontalOffset プロパティ)	R/W	印字開始位置における、X 座標のオフセットを示します。
14	印刷後動作の設定/取得 (MediaHandling プロパティ)	R/W	印刷後のポーズ、カッター、剥離の動作を示します。
15	開始オフセットの設定/取得 (StartOffset プロパティ)	R/W	印字開始位置の設定を示します。
16	終了オフセットの設定/取得 (StopOffset プロパティ)	R/W	印刷後のフィード量を示します。
17	センサー選択(透過/反射/なし)の設定/取得 (LabelSensor プロパティ)	R/W	用紙センサーを示します。
18	印字方法の設定/取得 (PrintMethod プロパティ)	R/W	印字方法(感熱、熱転写)の選択を示します。
19	センサー選択(前方/後方)の設定/取得 (SensorLocation プロパティ)	R/W	前方センサー、後方センサーの選択を示します。
20	ステータス(コマンドインタプリタ動作中)の取得 (CommandInterpreterInAction プロパティ)	R	コマンド処理中のステータスを示します。
21	ステータス(ペーパーエラー)の取得 (PaperError プロパティ)	R	ペーパーエラーのステータスを示します。
22	ステータス(リボンエンド)の取得 (RibbonEnd プロパティ)	R	リボンエンドのステータスを示します。
23	ステータス(バッチ処理印字中)の取得 (BatchProcessing プロパティ)	R	バッチ処理印字中のステータスを示します。
24	ステータス(印字中)の取得 (Printing プロパティ)	R	印字中のステータスを示します。
25	ステータス(ポーズ中)の取得 (Pause プロパティ)	R	ポーズ中のステータスを示します。
26	ステータス(剥離待ち中)の取得 (WaitingForPeeling プロパティ)	R	剥離待ち中のステータスを示します。
27	ステータス(ヘッド低温)の取得 (PrintHeadLowTemp プロパティ)	R	ヘッド低温のステータスを示します。
28	ステータス(ヘッド切れ)の取得 (PrintHeadFailure プロパティ)	R	ヘッド切れのステータスを示します。
29	ステータス(ヘッドオーバーヒート)の取得 (PrintHeadOverheat プロパティ)	R	ヘッドオーバーヒートのステータスを示します。
30	ステータス(メカオープン中)の取得 (MechanismOpen プロパティ)	R	メカオープン中のステータスを示します。
31	ステータス(オートカッター異常)の取得 (AutoCutterError プロパティ)	R	オートカッター異常のステータスを示します。

32	ステータス(ファンモーターストップ)の取得 (FanMotorError プロパティ)	R	ファンモーターストップのステータスを示します。
33	ステータス(その他エラー発生中)の取得 (MiscError プロパティ)	R	その他エラー発生中のステータスを示します。

LabelDesign クラス メソッド一覧

No	機能	詳細
1	クラス生成 (コンストラクタ)	コンストラクタです。
2	テキスト描画(プリンターフォント) (DrawTextPtrFont メソッド)	プリンターのフォントを使用して、テキストを描画します。
3	テキスト描画(プリンターダウンロードフォント) (DrawTextDLFont メソッド)	プリンターにダウンロードしたフォントを使用して、テキストを描画します。
4	テキスト描画(PC フォント) (DrawTextPCFont メソッド)	PC のフォントを使用して、テキストを描画します。
5	画像描画(プリンター格納画像) (DrawNVBitmap メソッド)	プリンターに格納している画像を描画します。
6	画像描画(ローカル画像ファイル) (DrawBitmap メソッド)	PC 上の画像ファイルを描画します。
7	バーコード描画(1D) (DrawBarCode メソッド)	バーコードを描画します。
8	バーコード描画(UPS MaxiCode) (DrawMaxiCode メソッド)	
9	バーコード描画(PDF417) (DrawPDF417 メソッド)	
10	バーコード描画(DataMatrix) (DrawDataMatrix メソッド)	
11	バーコード描画(QRCode) (DrawQRCode メソッド)	
12	バーコード描画(Aztec) (DrawAztec メソッド)	
13	バーコード描画(GS1DataBar(RSS)) (DrawGS1DataBar メソッド)	
14	線の描画 (DrawLine メソッド)	線を描画します。
15	矩形の描画 (DrawRect メソッド)	矩形を描画します。
16	矩形の描画(塗りつぶし) (FillRect メソッド)	矩形(塗りつぶし)を描画します。
17	円の描画 (DrawCircle メソッド)	円を描画します。
18	円の描画(塗りつぶし) (FillCircle メソッド)	円(塗りつぶし)を描画します。
19	多角形の描画 (DrawPolygon メソッド)	多角形を描画します。
20	多角形の描画(塗りつぶし) (FillPolygon メソッド)	多角形(塗りつぶし)を描画します。
21	生デザインコマンド挿入 (EmbedRawDesignCommand メソッド)	デザインコマンドを挿入します。

2. ライブラリインターフェイス

本 SDK のインターフェイスを以下に示します。

2.1. 戻り値

以降に示すメソッドは、下記の値を返します。

戻り値	説明
CLS_SUCCESS (0)	正常終了。
CLS_E_CONNECTED (1001)	プリンターへ既に接続済みです。
CLS_E_DISCONNECT (1002)	プリンターへ接続していません。
CLS_E_NOTCONNECT (1003)	プリンターへ接続できませんでした。
CLS_E_CONNECT_NOTFOUND (1004)	プリンター接続後の対応機種確認に失敗しました。
CLS_E_USB_BIDIRECTIONAL (1010)	Windows ドライバーのステータス機能が有効でした。
CLS_E_ILLEGAL (1101)	サポートされていない処理または無効なパラメータ値です。
CLS_E_OFFLINE (1102)	プリンターがオフラインです。
CLS_E_NOEXIST (1103)	指定のファイルが存在しません。
CLS_E_FAILURE (1104)	要求された処理が実行できません。
CLS_E_TIMEOUT (1105)	所定の時間が経過してもプリンターからの応答がありません。
CLS_E_NO_LIST (1106)	プリンター検索にてプリンターが見つかりません。
CLS_EPTR_PAPER_EMPTY (1202)	用紙切れエラーです。
CLS_EPTR_BADFORMAT (1203)	指定されたファイルの書式がサポートされていません。

2.2. LabelPrinter クラス

2.2.1. コンストラクタ

形式

LabelPrinter ()

パラメータ

ありません。

説明

Windows ライブラリのコンストラクタです。インスタンスを生成します。

戻り値

ありません。

使用例

```
LabelPrinter printer = new LabelPrinter();
```

2.2.2. Connect メソッド

形式

- 1) int Connect (int connectType, String address)
- 2) int Connect (int connectType, String address, int port)
- 3) int Connect (int connectType, String address, int port, int timeout)

パラメータ

パラメータの意味と設定可能な値は以下の通りです。

値	[IN/OUT]	意味	設定可能範囲
connectType	[IN]	接続タイプ(定数)	CLS_PORT_NET CLS_PORT_USB CLS_PORT_COM CLS_PORT_LPT CLS_PORT_Bluetooth
address	[IN]	接続先の IP アドレス、 または USB インターフェースポート 名 (Virtual printer port for USB)、 または COM、LPT のポート名、ま たは、Bluetooth デバイスアドレス	NET : 0.0.0.0~255.255.255.255 USB : USB001~ COM : COM1:~ LPT : LPT1:~ Bluetooth : 00:00:00:00:00:00~FF:FF:FF:FF:FF:FF
port	[IN]	接続先ポート番号	
timeout	[IN]	タイムアウト (msec)	

説明

このメソッドは、プリンターと接続するために使用します。プリンターの接続タイプとアドレスを指定してください。接続先ポート番号は、接続タイプに CLS_PORT_NET を指定した場合のみ有効です。省略された場合は、9100 番で接続します。

タイムアウトは、プリンターへの接続タイムアウト時間(ミリ秒単位)を指定します。省略された場合は、CLS_PORT_NET が 4000 ミリ秒、CLS_PORT_Bluetooth が 8000 ミリ秒でタイムアウトします。

プリンターと接続した際に、プリンターの対応機種を同時に確認します。

プリンターとの通信が不要になった場合は、必ず [Disconnect メソッド](#)を実行し、プリンターとの接続を切断してください。切断しなかった場合は、次の接続がエラーとなります。

戻り値

成功時は CLS_SUCCESS(0) を返します。失敗時は下記のエラーコードの説明を確認してください。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

エラーコード	説明
CLS_E_CONNECTED (1001)	既にプリンターと接続済みです。
CLS_E_NOTCONNECT (1003)	プリンターへ接続できませんでした。 ①プリンターが未接続 ②プリンターの電源が入っていない ③インターフェースポートのハンドルを取得できない
CLS_E_CONNECT_NOTFOUND (1004)	プリンター接続後の対応機種確認に失敗しました。 ①対応機種でない
CLS_E_USB_BIDIRECTIONAL (1010)	Windows ドライバーのステータス機能が有効でした。 Windows ドライバーの設定を変更してください(「 1.2 動作環境 」の「Windows ドライバーインストール」を参照)。

使用例

```
printer.Connect(LabelConst.CLS_PORT_NET, "192.168.129.130");  
  
printer.Connect(LabelConst.CLS_PORT_USB, "USB003");  
  
printer.Connect(LabelConst.CLS_PORT_COM, "COM1:");  
  
printer.Connect(LabelConst.CLS_PORT_LPT, "LPT1:");  
  
printer.Connect(LabelConst.CLS_PORT_Bluetooth, "00:01:90:DF:C1:1B");
```

2.2.3. Disconnect メソッド

形式

int Disconnect ()

パラメータ

ありません。

説明

このメソッドは、プリンターとの接続を切断するために使用します。

印刷の終了、あるいは、何らかのエラーが発生した場合は、本メソッドを実行して接続を切断してください。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
printer.Disconnect();
```

2.2.4. SetCommProperties メソッド

形式

int SetCommProperties(int baudRate, int parity, int handShake)

パラメータ

パラメータの意味と設定可能な値は以下の通りです。

値	[IN/OUT]	意味	設定可能範囲
baudRate	[IN]	ボーレート	CLS_COM_BAUDRATE_1200 CLS_COM_BAUDRATE_2400 CLS_COM_BAUDRATE_4800 CLS_COM_BAUDRATE_9600 (デフォルト) CLS_COM_BAUDRATE_19200 CLS_COM_BAUDRATE_38400 CLS_COM_BAUDRATE_57600 CLS_COM_BAUDRATE_115200
parity	[IN]	パリティ	CLS_COM_PARITY_NONE (デフォルト) CLS_COM_PARITY_ODD CLS_COM_PARITY_EVEN
handShake	[IN]	フロー制御	CLS_COM_HANDSHAKE_DTRDSR (デフォルト) CLS_COM_HANDSHAKE_XONXOFF

説明

このメソッドは、端末側のシリアルポートを設定するために使用します。

プリンター側と端末側のシリアルポート設定が同等でないと通信することができません。プリンターのシリアルポート設定についてはプリンターの取扱説明書をご参照ください。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
printer.SetCommProperties( LabelConst.CLS_COM_BAUDRATE_9600,
                          LabelConst.CLS_COM_PARITY_NONE,
                          LabelConst.CLS_COM_HANDSHAKE_DTRDSR );
printer.Connect( LabelConst.CLS_PORT_COM, "COM1:" );
```

2.2.5. PrinterCheck メソッド

形式

int PrinterCheck ()

パラメータ

ありません。

説明

このメソッドは、プリンターのステータスを取得し、ステータス情報を [CommandInterpreterInAction](#)、[PaperError](#)、[RibbonEnd](#)、[BatchProcessing](#)、[Printing](#)、[Pause](#)、[WaitingForPeeling](#)、[PrintHeadLowTemp](#)、[PrintHeadFailure](#)、[PrintHeadOverheat](#)、[MechanismOpen](#)、[AutoCutterError](#)、[FanMotorError](#)、[MiscError](#) プロパティに格納します。本メソッドの実行結果が失敗の場合は、通信異常やデバイスの異常が発生した可能性があります。この場合、[Disconnect メソッド](#)および [Connect メソッド](#)を使用して再接続してください。

接続後に時間を空けて印刷する場合は、必ず事前に本メソッドを実行し、プロパティを取得してプリンターのステータスを確認してください。

ネットワーク接続の場合、長時間放置すると自動的に切断されます。接続を保持する場合は、定期的に本メソッドを実行してください。

ネットワーク接続の場合、CL-S5xx/6xx/70x 系のプリンターは、予め「Parallel Error Output」の設定を OFF に変更しておいてください。パラレル接続の場合、全機種において「Parallel Error Output」の設定を OFF に変更しておいてください。設定の変更は、ラベルプリンターユーティリティの「拡張」タブから行うことができます。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
// PrinterCheck
if (LabelConst.CLS_SUCCESS == printer.PrinterCheck())
{
    // CommandInterpreterInActionプロパティ
    if ( printer.GetCommandInterpreterInAction() == 1 ) {
        // Command interpreter in action
    }
    // PaperErrorプロパティ
    if ( printer.GetPaperError() == 1 ) {
        // Paper error
    }
    // RibbonEndプロパティ
    if ( printer.GetRibbonEnd() == 1 ) {
        // Ribbon end
    }
    // BatchProcessingプロパティ
    if ( printer.GetBatchProcessing() == 1 ) {
        // Batch processing
    }
    // Printingプロパティ
    if ( printer.GetPrinting() == 1 ) {
        // Printing
    }
    // Pauseプロパティ
    if ( printer.GetPause() == 1 ) {
```

```
        // Pause
    }
    // WaitingForPeelingプロパティ
    if ( printer.GetWaitingForPeeling() == 1 ) {
        // Waitiong for peeling
    }
    // PrintHeadLowTempプロパティ
    if ( printer.GetPrintHeadLowTemp() == 1 ) {
        // Print head low temp
    }

    // PrintHeadFailureプロパティ
    if ( printer.GetPrintHeadFailure() == 1 ) {
        // Print head failure
    }
    // PrintHeadOverheatプロパティ
    if ( printer.GetPrintHeadOverheat() == 1 ) {
        // Print head overheat
    }
    // MechanismOpenプロパティ
    if ( printer.GetMechanismOpen() == 1 ) {
        // Mechanism open
    }
    // AutoCutterErrorプロパティ
    if ( printer.GetAutoCutterError() == 1 ) {
        // Auto cutter error
    }
    // FanMotorErrorプロパティ
    if ( printer.GetFanMotorError() == 1 ) {
        // Fan motor error
    }
    // MiscErrorプロパティ
    if ( printer.GetMiscError() == 1 ) {
        // Misc error
    }
}
else
{
    // Fail
}
```


2.2.6. Print メソッド

形式

int Print (LabelDesign design, int quantity)

パラメータ

パラメータの意味と設定可能な値は以下の通りです。

値	[IN/OUT]	意味	設定可能範囲
design	[IN]	LabelDesign クラスインスタンス	
quantity	[IN]	印刷枚数	1～9999

説明

このメソッドは、デザインしたラベルを印刷する際に使用します。プリンターへ印刷要求を送信します。印刷要求を行う際、LabelDesign クラスで指定されたラベルのフォーマットや、プロパティの設定をコマンドに反映します。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
design.FillCircle(50, 50, 50, LabelConst.CLS_SHADED_PTN_11);  
printer.Print(design, 0001);
```

2.2.7. Preview メソッド

形式

```
int Preview (LabelDesign design, int resolution, int measurementUnit, int paperWidth,
            int paperHeight)
```

パラメータ

パラメータの意味と設定可能な値は以下の通りです。

値	[IN/OUT]	意味	設定可能範囲
design	[IN]	LabelDesign クラスインスタンス	
resolution	[IN]	解像度(dpi)	CLS_PRT_RES_203(203dpi) CLS_PRT_RES_300(300dpi)
measurementUnit	[IN]	単位	CLS_UNIT_MILLI CLS_UNIT_INCH
paperWidth	[IN]	ラベル幅	「単位」の設定により、以下の範囲で設定が可能です。 CLS_UNIT_MILLI: 254～8128 CLS_UNIT_INCH: 100～3200
paperHeight	[IN]	ラベル長	

説明

このメソッドは、デザインしたラベルをプレビューする際に使用します。プレビューには、LabelDesign クラスでデザインしたオブジェクトが反映されます。

解像度、単位の設定値は、プリンターと同じ設定としてください。プリンターの設定と異なっていた場合、オブジェクトの印字位置にズレが生じることがあります。

ラベル幅とラベル長は、使用しているラベル用紙のラベルサイズを設定してください。ラベル幅とラベル長で設定されたサイズが、プレビューの表示サイズに反映されます。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
design.FillCircle(50, 50, 50, LabelConst.CLS_SHADED_PTN_11);
printer.Preview(design, LabelConst.CLS_PRT_RES_203,
               LabelConst.CLS_UNIT_INCH, 300, 400);
```

2.2.8. StoreNVBitmap メソッド

形式

- 1) int StoreNVBitmap (String filePath, String name, int rotation, int width, int height)
- 2) int StoreNVBitmap (String filePath, String name, int rotation, int width, int height, int mode)
- 3) int StoreNVBitmap (Bitmap bitmap, String name, int rotation, int width, int height)
- 4) int StoreNVBitmap (Bitmap bitmap, String name, int rotation, int width, int height, int mode)

パラメータ

パラメータの意味と設定可能な値は以下の通りです。

値	[IN/OUT]	意味	設定可能範囲
filePath	[IN]	対象ファイル名	
bitmap	[IN]	ビットマップ型データ	
name	[IN]	格納名称	以下の条件を除いた ASCII コードが使用可能 ・先頭がアンダースコアは不可 ・以下の文字は使用不可(禁則文字) ¥ / : * ? " < >
rotation	[IN]	回転方向(定数)	CLS_RT_NORMAL: 回転なし CLS_RT_RIGHT90: 右 90 度回転 CLS_RT_ROTATE180: 右 180 度回転 CLS_RT_LEFT90: 左 90 度回転
width	[IN]	横サイズ(ピクセル単位)	
height	[IN]	縦サイズ(ピクセル単位)	
mode	[IN]	ビットマップモード	CLS_BM_MODE_HT_THRESHOLD: ハーフトーン しきい値 CLS_BM_MODE_HT_DITHER: ハーフトーン ディザー

説明

このメソッドは、ビットマップのファイル名またはビットマップ型データ、格納名称、回転方向、横サイズ、縦サイズ、モードを指定して、プリンターのフラッシュメモリにビットマップ画像(ロゴ)を登録するために使用します。登録したロゴは、[DrawNVBitmap メソッド](#)を使用して描画できます。登録可能なビットマップ形式は、BMP/GIF/EXIF/JPG/PNG/TIFF です。

登録されるビットマップ画像サイズは、指定された縦横サイズに合わせて、アスペクト比を維持した状態でリサイズします。

例: 以下の場合、リサイズ後の画像サイズは「200x50」ピクセルとなる。

画像サイズ: 400x100 ピクセル

横サイズ: 200

縦サイズ: 200

指定のサイズが 0 の場合、対象の画像サイズを基準とする。

例: 以下の場合、リサイズ後の画像サイズは「800x200」ピクセルとなる。

画像サイズ: 400x100 ピクセル

横サイズ: 0

縦サイズ: 200

モードは、ハーフトーン処理方法を指定します。モードが省略された場合は、CLS_BM_MODE_HT_DITHER で登録します。モードに関する詳細は、次の通りです。

ハーフトーン ハーフトーン処理方法を指定します。

設定値	説明
CLS_BM_MODE_HT_THRESHOLD	しきい値 文字の印刷に適します。
CLS_BM_MODE_HT_DITHER	ディザー グラフィックの印刷に適します。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
// 画像登録(プリンターへの格納)
printer.StoreNVBitmap("C:¥¥TestFolder¥¥Sample.bmp", "Sample",
    LabelConst.CLS_RT_NORMAL, 0, 0);

// 画像描画(プリンター側格納画像)
design.DrawNVBitmap("Sample", 1, 1, 0, 0);

// 印刷
printer.Print(design, 0001);
```

2.2.9. ClearOutput メソッド

形式

int ClearOutput ()

パラメータ

ありません。

説明

プリンターをリセットします。プリンター側では、電源 ON 時と同等の初期化処理が行われるため、未印字データもクリアされます。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
printer.ClearOutput();
```

2.2.10. SendData メソッド

形式

int SendData (byte[] data)

パラメータ

パラメータの意味と設定可能な値は以下の通りです。

値	[IN/OUT]	意味	設定可能範囲
data	[IN]	送信データ	

説明

このメソッドは、バイトデータをそのままプリンターに送信するために使用します。
通常は必要ありませんが、プリンターのコマンドを直接送信したい場合に使用します。
ご使用の際は、他のメソッドに影響を与えない様に注意する必要があります。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
byte[] Data = new byte[] {0x01, 0x23}; // [01]# (リセット)

printer.SendData(Data);
```

2.2.11. SearchCitizenPrinter メソッド

形式

CitizenPrinterInfo[] SearchCitizenPrinter (int ifType, int searchTime, out int result)

パラメータ

パラメータの意味と設定可能な値は以下の通りです。

値	[IN/OUT]	意味	設定可能範囲
ifType	[IN]	接続インターフェース(定数)	CLS_PORT_NET CLS_PORT_USB CLS_PORT_COM CLS_PORT_LPT CLS_PORT_Bluetooth
searchTime	[IN]	検索タイムアウト時間(秒)	1～30
result	[OUT]	エラーコード	

result には成功時は CLS_SUCCESS (0) を返します。失敗時は下記のエラーコードの説明を確認してください。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

値	説明
CLS_E_ILLEGAL (1101)	無効なパラメータ値です。 ①未対応の接続タイプを指定された場合 ②範囲外の検索時間を指定された場合
CLS_E_NO_LIST (1106)	検索の結果、プリンターが発見出来ませんでした。

説明

このメソッドは、プリンターを検索し、プリンターの情報リストを取得するために使用します。接続タイプと検索時間を指定して実行してください。検索時間経過後に、結果を result パラメータにセットし、検索できたプリンターの情報を配列で返します。

接続タイプが CLS_PORT_NET の場合、CL-S400/520/530/620/630/700/703 系のプリンターと、CL-E720/730/300/303/321/331 系のプリンターのみ検索可能です。検索時間の推奨値は 3 秒以上です。これより短い時間の場合、ネットワークの状態によっては検索漏れが発生する場合があります。

接続タイプが CLS_PORT_USB の場合、接続されている CITIZEN 製プリンターを検索します。検索時間である searchTime に関係無く、検索は直ぐに完了します。

接続タイプが CLS_PORT_COM または CLS_PORT_LPT の場合、利用可能なポートを検索します。検索時間である searchTime に関係無く、検索は直ぐに完了します。

接続タイプが CLS_PORT_Bluetooth の場合、検索時間に 0 を指定すると、ペアリング済みのデバイスを取得します。検索時間に 1～30 を指定すると、接続可能なデバイスを取得します。検索時間の推奨値は 10 秒以上です。これより短い時間の場合、Bluetooth の状態によっては検索漏れが発生する場合があります。

戻り値

検索が成功した場合はプリンターの情報リストが返されます。失敗した場合、空のリストを返します。

プリンターの情報リストは CitizenPrinterInfo 型で格納されており、接続タイプによって取得できる情報が異なります。

接続タイプ	CitizenPrinterInfo	取得できる情報
CLS_PORT_NET	ipAddress	IP アドレス
	macAddress	MAC アドレス
	bdAddress	(空)
	deviceName	(空)
	printerModel	(空)

CLS_PORT_USB	ipAddress	(空)
	macAddress	(空)
	bdAddress	(空)
	deviceName	USB インターフェースポート名
	printerModel	プリンターモデル名
CLS_PORT_COM CLS_PORT_LPT	ipAddress	(空)
	macAddress	(空)
	bdAddress	(空)
	deviceName	ポート名
	printerModel	(空)
CLS_PORT_Bluetooth	ipAddress	(空)
	macAddress	(空)
	bdAddress	Bluetooth デバイスアドレス
	deviceName	Bluetooth デバイス名
	printerModel	(空)

使用例

```
int errcode;

CitizenPrinterInfo[] list = printer.SearchCitizenPrinter(
    LabelConst.CLS_PORT_NET, 3, out errcode);

for (int i = 0; i < list.Length; i++)
{
    Console.WriteLine("IP Address: " + list[i].ipAddress);
    Console.WriteLine("MAC Address: " + list[i].macAddress);
}
```


2.2.12. SearchLabelPrinter メソッド

形式

string[] SearchLabelPrinter (int ifType, int searchTime, out int result)

パラメータ

パラメータの意味と設定可能な値は以下の通りです。

値	[IN/OUT]	意味	設定可能範囲
ifType	[IN]	接続タイプ(定数)	CLS_PORT_NET CLS_PORT_USB CLS_PORT_COM CLS_PORT_LPT CLS_PORT_Bluetooth
searchTime	[IN]	検索時間 (秒)	1～30
result	[OUT]	エラーコード	

result には成功時は CLS_SUCCESS(0) を返します。失敗時は下記のエラーコードの説明を確認してください。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

値	説明
CLS_E_ILLEGAL (1101)	無効なパラメータ値です。 ①未対応の接続タイプを指定された場合 ②範囲外の検索時間を指定された場合
CLS_E_NO_LIST (1106)	検索の結果、プリンターが発見出来ませんでした。

説明

このメソッドは、プリンターを検索し、アドレスのリストを取得するために使用します。接続タイプと検索時間を指定して実行してください。検索時間経過後に、結果を result パラメータにセットし、検索できたプリンターのアドレスを string 配列で返します。

接続タイプが CLS_PORT_NET の場合、CL-S400/520/530/620/630/700/703 系のプリンターと、CL-E720/730/300/303/321/331 系のプリンターのみ検索可能です。検索時間の推奨値は 3 秒以上です。これより短い時間の場合、ネットワークの状態によっては検索漏れが発生する場合があります。

接続タイプが CLS_PORT_USB の場合、端末に接続されている CITIZEN 製プリンターを検索します。検索時間である searchTime に関係無く、検索は直ぐに完了します。

接続タイプが CLS_PORT_COM または CLS_PORT_LPT の場合、利用可能なポートを検索します。検索時間である searchTime に関係無く、検索は直ぐに完了します。

接続タイプが CLS_PORT_Bluetooth の場合、検索時間に 0 を指定すると、ペアリング済みのアドレスを取得します。検索時間に 1～30 を指定すると、接続可能なアドレスを取得します。検索時間の推奨値は 10 秒以上です。これより短い時間の場合、Bluetooth の状態によっては検索漏れが発生する場合があります。

戻り値

検索が成功した場合、プリンターのアドレスのリストを返します。失敗した場合、空のリストを返します。

使用例

```
int errcode;

string[] list = printer.SearchLabelPrinter(
    LabelConst.CLS_PORT_NET, 3, out errcode);
```

2.2.13. PrinterCheckEx メソッド

形式

- 1) int PrinterCheckEx (int connectType, String address)
- 2) int PrinterCheckEx (int connectType, String address, int port)
- 3) int PrinterCheckEx (int connectType, String address, int port, int timeout)

パラメータ

パラメータの意味と設定可能な値は以下の通りです。

値	[IN/OUT]	意味	設定可能範囲
connectType	[IN]	接続タイプ(定数)	CLS_PORT_NET CLS_PORT_USB CLS_PORT_COM CLS_PORT_LPT CLS_PORT_Bluetooth
address	[IN]	接続先の IP アドレス、 または USB インターフェースポート 名 (Virtual printer port for USB)、 または COM、LPT のポート名、ま たは、Bluetooth デバイスアドレス	NET : 0.0.0.0~255.255.255.255 USB : USB001~ COM : COM1:~ LPT : LPT1:~ Bluetooth : 00:00:00:00:00:00~FF:FF:FF:FF:FF:FF
port	[IN]	接続先ポート番号	
timeout	[IN]	タイムアウト (msec)	

説明

このメソッドは、プリンターへ接続後に、プリンターのステータスを取得し、ステータス情報を [CommandInterpreterInAction](#)、[PaperError](#)、[RibbonEnd](#)、[BatchProcessing](#)、[Printing](#)、[Pause](#)、[WaitingForPeeling](#)、[PrintHeadLowTemp](#)、[PrintHeadFailure](#)、[PrintHeadOverheat](#)、[MechanismOpen](#)、[AutoCutterError](#)、[FanMotorError](#)、[MiscError](#) プロパティに格納します。処理が完了した後は接続を切断します。

このメソッドは、プリンターの状態にかかわらず接続して処理ができます。

ネットワーク接続の場合、CL-S5xx/6xx/70x 系のプリンターは、予め「Parallel Error Output」の設定を OFF に変更しておいてください。パラレル接続の場合、全機種において「Parallel Error Output」の設定を OFF に変更しておいてください。設定の変更は、ラベルプリンターユーティリティの「拡張」タブから行うことができます。

戻り値

成功時は CLS_SUCCESS(0) を返します。失敗時は下記のエラーコードの説明を確認してください。

それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

エラーコード	説明
CLS_E_CONNECTED (1001)	既にプリンターと接続済みです。
CLS_E_NOTCONNECT (1003)	プリンターへ接続できませんでした。 ①プリンターが未接続 ②プリンターの電源が入っていない ③インターフェースポートのハンドルを取得できない
CLS_E_FAILURE (1104)	要求された処理が実行できません。 シリアル接続とパラレル接続の場合は、プリンターが未接続の場合も含まれます。
CLS_E_USB_BIDIRECTIONAL (1010)	Windows ドライバーのステータス機能が有効でした。 Windows ドライバーの設定を変更してください(「 1.2 動作環境 」の「Windows ドライバーインストール」を参照)。

使用例

```
// PrinterCheckEx
if (LabelConst.CLS_SUCCESS ==
    printer.PrinterCheckEx(LabelConst.CLS_PORT_USB, "USB003"))
{
    // CommandInterpreterInActionプロパティ
    if ( printer.GetCommandInterpreterInAction() == 1 ) {
        // Command interpreter in action
    }
    // PaperErrorプロパティ
    if ( printer.GetPaperError() == 1 ) {
        // Paper error
    }
    // RibbonEndプロパティ
    if ( printer.GetRibbonEnd() == 1 ) {
        // Ribbon end
    }
    // BatchProcessingプロパティ
    if ( printer.GetBatchProcessing == 1 ) {
        // Batch processing
    }
    // Printingプロパティ
    if ( printer.GetPrinting() == 1 ) {
        // Printing
    }
    // Pauseプロパティ
    if ( printer.GetPause() == 1 ) {
        // Pause
    }
    // WaitingForPeelingプロパティ
    if ( printer.GetWaitingForPeeling() == 1 ) {
        // Waitiong for peeling
    }
    // PrintHeadLowTempプロパティ
    if ( printer.GetPrintHeadLowTemp() == 1 ) {
        // Print head low temp
    }

    // PrintHeadFailureプロパティ
    if ( printer.GetPrintHeadFailure() == 1 ) {
        // Print head failure
    }
    // PrintHeadOverheatプロパティ
    if ( printer.GetPrintHeadOverheat() == 1 ) {
        // Print head overheat
    }
    // MechanismOpenプロパティ
    if ( printer.GetMechanismOpen() == 1 ) {
        // Mechanism open
    }
    // AutoCutterErrorプロパティ
    if ( printer.GetAutoCutterError() == 1 ) {
        // Auto cutter error
    }
    // FanMotorErrorプロパティ
    if ( printer.GetFanMotorError() == 1 ) {
```

```
        // Fan motor error
    }
    // MiscErrorプロパティ
    if ( printer.GetMiscError() == 1 ) {
        // Misc error
    }
}
else
{
    // Fail
}
```

2.2.14. ClearOutputEx メソッド

形式

- 4) int ClearOutputEx (int connectType, String address)
- 5) int ClearOutputEx (int connectType, String address, int port)
- 6) int ClearOutputEx (int connectType, String address, int port, int timeout)

パラメータ

パラメータの意味と設定可能な値は以下の通りです。

値	[IN/OUT]	意味	設定可能範囲
connectType	[IN]	接続タイプ(定数)	CLS_PORT_NET CLS_PORT_USB CLS_PORT_COM CLS_PORT_LPT CLS_PORT_Bluetooth
address	[IN]	接続先の IP アドレス、 または USB インターフェースポート 名 (Virtual printer port for USB)、 または COM、LPT のポート名、ま たは、Bluetooth デバイスアドレス	NET : 0.0.0.0~255.255.255.255 USB : USB001~ COM : COM1:~ LPT : LPT1:~ Bluetooth : 00:00:00:00:00:00~FF:FF:FF:FF:FF:FF
port	[IN]	接続先ポート番号	
timeout	[IN]	タイムアウト (msec)	

説明

このメソッドは、プリンターへ接続後に、プリンターをリセットします。プリンター側では、電源 ON 時と同等の初期化処理が行われるため、未印字データもクリアされます。処理が完了した後は接続を切断します。
このメソッドは、プリンターの状態にかかわらず接続して処理ができます。

戻り値

成功時は CLS_SUCCESS(0) を返します。失敗時は下記のエラーコードの説明を確認してください。
それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

エラーコード	説明
CLS_E_CONNECTED (1001)	既にプリンターと接続済みです。
CLS_E_NOTCONNECT (1003)	プリンターへ接続できませんでした。 ①プリンターが未接続 ②プリンターの電源が入っていない ③インターフェースポートのハンドルを取得できない
CLS_E_FAILURE (1104)	要求された処理が実行できません。 シリアル接続とパラレル接続の場合は、プリンターが未接続の場合も含まれます。
CLS_E_USB_BIDIRECTIONAL (1010)	Windows ドライバーのステータス機能が有効でした。 Windows ドライバーの設定を変更してください(「 1.2 動作環境 」の「Windows ドライバーインストール」を参照)。

使用例

```
printer.ClearOutputEx(LabelConst.CLS_PORT_USB, "USB003");
```

2.2.15. SetLog メソッド

形式

void SetLog (int mode, string path, int maxSize)

パラメータ

パラメータの意味と設定可能な値は以下の通りです。

値	[IN/OUT]	意味	設定可能範囲
mode	[IN]	ログモード	0 : 記録なし 1 : アクセス履歴の記録 2 : エラーのみ記録
path	[IN]	格納フォルダ	
maxSize	[IN]	ログサイズ	0: サイズ制限なし 1～: 最大サイズ(MB)

説明

このメソッドは、ログ機能を設定するために使用します。ログ機能の詳細は「[3.2 ログ機能について](#)」を参照してください。

戻り値

成功時は CLS_SUCCESS (0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
printer.SetLog(1, "C:¥¥LOG", 10);
```

2.2.16. HorizontalMagnification プロパティ

形式

int HorizontalMagnification

属性

Read/Write

説明

このプロパティは、水平方向のピクセルサイズ(ドット構成単位)を保持します。値は、1(dot)もしくは、2(dot)で指定します。

設定方法

int SetHorizontalMagnification (int HorizontalMagnification)

パラメータに、設定したいプロパティ値を指定してください。

デフォルトを「1」とし、未設定時はデフォルトでコマンドを送信します。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetHorizontalMagnification ()

戻り値として、設定されている水平ピクセルサイズを返します。

デフォルトを「1」とし、未設定時はデフォルトが返ります。

使用例

```
int pixelsize;
```

```
pixelsize = printer.GetHorizontalMagnification();  
printer.SetHorizontalMagnification(2);
```

2.2.17. VerticalMagnification プロパティ

形式

int VerticalMagnification

属性

Read/Write

説明

このプロパティは、垂直方向のピクセルサイズ(dot 構成単位)を保持します。値は、1(dot)から 3(dot)の間で指定します。

設定方法

int SetVerticalMagnification (int VerticalMagnification)

パラメータに、設定したいプロパティ値を指定してください。

デフォルトを「1」とし、未設定時はデフォルトでコマンドを送信します。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetVerticalMagnification ()

戻り値として、設定されている垂直ピクセルサイズを返します。

デフォルトを「1」とし、未設定時はデフォルトが返ります。

使用例

```
int pixelsize;
```

```
printer.SetVerticalMagnification(3);  
pixelsize = printer.GetVerticalMagnification();
```


2.2.18. FormatAttribute プロパティ

形式

int FormatAttribute

属性

Read/Write

説明

このプロパティは、印刷領域が重なっている箇所の展開方法を保持します。値は、以下の通りです。

0: XOR 展開指定となり、文字やバーコードの重なった部分が白抜きとなります。(初期値)

1: OR 展開指定となり、文字やバーコードの重ね書きを行います。

設定方法

int SetFormatAttribute (int FormatAttribute)

パラメータに、設定したいプロパティ値を指定してください。

未設定時は、コマンドを送信しません。

プリンターのデフォルト値は「0」です。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetFormatAttribute ()

戻り値として、設定されている展開方法を返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int attr;

printer.SetFormatAttribute(1);
attr = printer.GetFormatAttribute();
```

印刷結果



2.2.19. ContinuousMediaLength プロパティ

形式

int ContinuousMediaLength

属性

Read/Write

説明

連続紙を使用した場合のラベル長の設定です。
ラベルフォーマットの長さは、このコマンドで設定した長さになります。
オートカッター使用時は、この設定の長さで、ラベルカットを行います。

インチ設定	0001 ~ 9999 (0.01 インチ～99.99 インチ)
ミリ設定	0001 ~ 9999 (0.1mm～999.9mm)

設定方法

int SetContinuousMediaLength (int ContinuousMediaLength)

パラメータに、設定したいプロパティ値を指定してください。
未設定時は、コマンドを送信しません。
プリンターのデフォルト値は「0000」です。
成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetContinuousMediaLength ()

戻り値として、設定されている連続用紙長を返します。
未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int medialength;

printer.SetContinuousMediaLength(2000);
medialength = printer.GetContinuousMediaLength();
```

2.2.20. MeasurementUnit プロパティ

形式

int MeasurementUnit

属性

Read/Write

説明

このプロパティは、全ての全距離指定コマンドのパラメータの単位を保持します。値は以下の通りです。

値	意味
CLS_UNIT_MILLI (0)	ミリ(0.1mm 単位)
CLS_UNIT_INCH (1)	インチ(0.01 インチ単位)

設定方法

int SetMeasurementUnit (int MeasurementUnit)

パラメータに、設定したいプロパティ値を指定してください。

未設定時は、コマンドを送信しません。

プリンターのデフォルト値は「1」です。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetMeasurementUnit ()

戻り値として、設定されている距離指定パラメータの単位を返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int unit;
```

```
printer.SetMeasurementUnit (LabelConst.CLS_UNIT_MILLI);  
unit = printer.GetMeasurementUnit();
```

2.2.21. PrintSpeed プロパティ

形式

int PrintSpeed

属性

Read/Write

説明

このプロパティは、印字部分の速度を保持します。値は、1 文字のアルファベットか数字で設定します。(「[3.3. パラメータ](#)」の No21 を参照)

※指定可能範囲、プリンターの初期値は、機種により異なります。

使用機種の取扱い説明書をご参照ください。

設定方法

int SetPrintSpeed (int PrintSpeed)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetPrintSpeed ()

戻り値として、設定されている印刷速度を返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int printspeed;  
  
printer.SetPrintSpeed(LabelConst.CLS_SPEEDSETTING_X);  
printspeed = printer.GetPrintSpeed();
```

2.2.22. FeedSpeed プロパティ

形式

int FeedSpeed

属性

Read/Write

説明

このプロパティは、非印字部分の速度を保持します。値は、1 文字のアルファベットか数字で設定します。

(「[3.3.パラメータ](#)」の No21 を参照)

※指定可能範囲、プリンターの初期値は、機種により異なります。

使用機種の取扱い説明書をご参照ください。

設定方法

int SetFeedSpeed (int FeedSpeed)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetFeedSpeed ()

戻り値として、設定されている非印刷部分の紙送り速度を返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int feedspeed;
```

```
printer.SetFeedSpeed(LabelConst.CLS_SPEEDSETTING_W);  
feedspeed = printer.GetFeedSpeed();
```

2.2.23. SlewSpeed プロパティ

形式

int SlewSpeed

属性

Read/Write

説明

このプロパティは、紙送りのフィード速度を保持します。値は、1 文字のアルファベットか数字で設定します。

(「[3.3.パラメータ](#)」の No21 を参照)

※指定可能範囲、プリンターの初期値は、機種により異なります。

使用機種の取扱い説明書をご参照ください。

設定方法

int SetSlewSpeed (int SlewSpeed)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetSlewSpeed ()

戻り値として、設定されている紙送り部分の紙送り速度を返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int slewspeed;
```

```
printer.SetSlewSpeed(LabelConst.CLS_SPEEDSETTING_V);  
slewspeed = printer.GetSlewSpeed();
```

2.2.24. BackupSpeed プロパティ

形式

int BackupSpeed

属性

Read/Write

説明

このプロパティは、バックフィード時の速度を保持します。値は、1 文字のアルファベットか数字で設定します。

値	意味
CLS_SPEEDSETTING_A or CLS_SPEEDSETTING_B	1.0 インチ (25.4mm) / 秒
CLS_SPEEDSETTING_C or CLS_SPEEDSETTING_D	2.0 インチ (50.8mm) / 秒
CLS_SPEEDSETTING_E or CLS_SPEEDSETTING_F	3.0 インチ (76.2mm) / 秒
CLS_SPEEDSETTING_G or CLS_SPEEDSETTING_H	4.0 インチ (101.6mm) / 秒
CLS_SPEEDSETTING_I or CLS_SPEEDSETTING_J	5.0 インチ (127.0mm) / 秒
CLS_SPEEDSETTING_K or CLS_SPEEDSETTING_L	6.0 インチ (152.4mm) / 秒
CLS_SPEEDSETTING_M or CLS_SPEEDSETTING_N	7.0 インチ (177.8mm) / 秒
CLS_SPEEDSETTING_O	8.0 インチ (203.2mm) / 秒
CLS_SPEEDSETTING_1 ～ CLS_SPEEDSETTING_8	1.0 インチ (25.4mm) / 秒 ～ 8.0 インチ (203.2mm) / 秒

※指定可能範囲、プリンターの初期値は、機種により異なります。

使用機種の取扱い説明書をご参照ください。

設定方法

int SetBackupSpeed (int BackupSpeed)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetBackupSpeed ()

戻り値として、設定されているバックフィード設定を返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int backupspeed;

printer.SetBackupSpeed(LabelConst.CLS_SPEEDSETTING_O);
backupspeed = printer.GetBackupSpeed();
```

2.2.25. PrintDarkness プロパティ

形式

int PrintDarkness

属性

Read/Write

説明

このプロパティは、印刷濃度の設定を保持します。設定範囲と初期値は以下の通りです。

設定範囲 0 ~ 30

初期値 10

設定方法

int SetPrintDarkness (int PrintDarkness)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetPrintDarkness ()

戻り値として、設定されている印字濃度を返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int printdarkness;  
  
printer.SetPrintDarkness(30);  
printdarkness = printer.GetPrintDarkness();
```


2.2.26. DoubleHeat プロパティ

形式

int DoubleHeat

属性

Read/Write

説明

このプロパティは、ダブルヒート設定を保持します。ダブルヒートを設定することで、同じ位置に 2 回熱をかけて印字します。そのため、印字速度は半分になりますが、より濃く印字することができます。

- 0: ダブルヒート OFF (初期値)
- 1: ダブルヒート ON

設定方法

int SetDoubleHeat (int DoubleHeat)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetDoubleHeat ()

戻り値として、設定されているダブルヒート設定を返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int doubleheat;  
  
printer.SetDoubleHeat(1);  
doubleheat = printer.GetDoubleHeat();
```

2.2.27. VerticalOffset プロパティ

形式

int VerticalOffset

属性

Read/Write

説明

印字内容全体の位置を調整する為に、紙の上下(行)方向の印字開始位置のオフセット値を設定します。

インチ設定	0000 ~ 9999 (0.00 インチ~99.99 インチ)
ミリ設定	0000 ~ 9999 (0.0mm~999.9mm)
初期値	0000

設定方法

int SetVerticalOffset (int VerticalOffset)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetVerticalOffset ()

戻り値として、設定されている上下オフセットを返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int verticaloffset;

printer.SetVerticalOffset(10);
verticaloffset = printer.GetVerticalOffset();
```

2.2.28. HorizontalOffset プロパティ

形式

int HorizontalOffset

属性

Read/Write

説明

印字内容全体の位置を調整する為、用紙の左右オフセット値(列方向の印字開始位置)を設定します。

インチ設定	0000 ~ 9999 (0.00 インチ~99.99 インチ)
ミリ設定	0000 ~ 9999 (0.0mm~999.9mm)
初期値	0000

設定方法

int SetHorizontalOffset (int HorizontalOffset)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetHorizontalOffset ()

戻り値として、設定されている左右オフセットを返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int horizontaloffset;

printer.SetHorizontalOffset(20);
horizontaloffset = printer.GetHorizontalOffset();
```

2.2.29. MediaHandling プロパティ

形式

int MediaHandling

属性

Read/Write

説明

このプロパティは、印刷後の動作設定を保持します。設定可能な動作は、以下の通りです。

値	意味
CLS_MEDIAHANDLING_NONE (0)	印刷後の動作を OFF にします。
CLS_MEDIAHANDLING_TEAROFF (1)	排出動作のみ ON にします。
CLS_MEDIAHANDLING_DISPENSES (2)	剥離センサーと排出動作を ON にします。
CLS_MEDIAHANDLING_PAUSE (3)	排出動作を ON にし、排出後にポーズ状態にします。
CLS_MEDIAHANDLING_CUT (4)	オートカッターのみ ON にします。
CLS_MEDIAHANDLING_CUTANDPAUSE (5)	オートカッターを ON にし、カット後にポーズ状態にします。
CLS_MEDIAHANDLING_PEELOFF (6)	剥離センサーのみ ON にします。
CLS_MEDIAHANDLING_REWIND (7)	リワインダー(巻き取り)モードを ON にします。

設定方法

int SetMediaHandling (int MediaHandling)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetMediaHandling ()

戻り値として、設定されている印刷後動作を返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int mediahandling;
```

```
printer.SetMediaHandling(LabelConst.CLS_MEDIAHANDLING_PAUSE);
mediahandling = printer.GetMediaHandling();
```

2.2.30. StartOffset プロパティ

形式

int StartOffset

属性

Read/Write

説明

基準点から印字ヘッドまでの距離を指定することができます。この値を変える事により、物理的な印字開始位置を変更することが出来ます。

インチ指定			ミリ指定		
初期値	最小値	最大値	初期値	最小値	最大値
0220	0120	0320	0559	0305	0813

※単位 0.01 インチ又は 0.1mm

設定方法

int SetStartOffset(int StartOffset)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

インチ/ミリ指定の対応のチェックは行いません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetStartOffset ()

戻り値として、設定されている開始オフセットを返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int startoffset;

printer.SetStartOffset(220);
startoffset = printer.GetStartOffset();
```

2.2.31. StopOffset プロパティ

形式

int StopOffset

属性

Read/Write

説明

下記範囲の値にて、基準点(基準線)からカット位置又は、剥離位置までの距離を指定できます。

指定値が小さい場合、フィード量が小さいので印刷したラベルをカットしてしまいます。

指定値が適切な場合、必要量フィード後、紙間でカットします。

指定値が大きい場合、フィード量が大きいので、次のラベルをカットしてしまいます。

動作	インチ指定			ミリ指定		
	初期値	最小値	最大値	初期値	最小値	最大値
通常印刷	0000	0000	9999	0000	0000	9999
カッター	0100			0254		
剥離	0050			0127		
ティアオフ	0070			0178		

※単位 0.01 インチ又は 0.1mm

設定方法

int SetStopOffset (int StopOffset)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetStopOffset ()

戻り値として、設定されている終了オフセットを返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int stopoffset;

printer.SetStopOffset(240);
stopoffset = printer.GetStopOffset();
```

2.2.32. LabelSensor プロパティ

形式

int LabelSensor

属性

Read/Write

説明

このプロパティは、用紙センサーの設定を保持します。設定値は、以下の通りです。

値	意味
CLS_SELSENSOR_NONE (0)	なし。 なしの場合、連続用紙長設定の値(ContinuousMediaLength プロパティ)をチェックし、未設定、または 0000 の場合は引数エラーとします。
CLS_SELSENSOR_SEETHROUGH (1)	エッジセンサー。 ラベル紙の紙間、ダイカット紙、タグ紙のノッチ穴検出などに使用します。
CLS_SELSENSOR_REFLECT (2)	反射型用紙センサー。 ラベル裏面に印刷された黒線を検出して、ラベル位置を認識します。

設定方法

int SetLabelSensor (int LabelSensor)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetLabelSensor ()

戻り値として、設定されている用紙センサーを返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
// センサー「なし」以外るとき
```

```
int labelsensor;
```

```
printer.SetLabelSensor(LabelConst.CLS_SELSENSOR_SEETHROUGH);  
labelsensor = printer.GetLabelSensor();
```

```
// センサー「なし」のとき
```

```
int labelsensor;
```

```
printer.SetContinuousMediaLength(100);  
printer.SetLabelSensor(LabelConst.CLS_SELSENSOR_NONE);  
labelsensor = printer.GetLabelSensor();
```

2.2.33. PrintMethod プロパティ

形式

int PrintMethod

属性

Read/Write

説明

リボンを使用する熱転写モードと感熱紙を使用する感熱モードの印刷方法の指定を行います。

値	意味
CLS_PRTMETHOD_TT (0)	熱転写モード
CLS_PRTMETHOD_DT (1)	感熱モード

設定方法

int SetPrintMethod (int PrintMethod)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetPrintMethod ()

戻り値として、設定されている印字方法を返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int printmethod;  
  
printer.SetPrintMethod(LabelConst.CLS_PRTMETHOD_DT);  
printmethod = printer.GetPrintMethod();
```


2.2.34. SensorLocation プロパティ

形式

int SensorLocation

属性

Read/Write

説明

前方センサーと後方センサーの 2 種類の紙検出センサーを搭載している機種において、使用する紙検出センサーを切り替えます。設定した内容はバックアップメモリに記憶され電源を切っても設定は有効です。

値	意味
CLS_SENS_LOCATION_FRONT (0)	前方センサー
CLS_SENS_LOCATION_ADJUSTABLE (1)	後方センサー

設定方法

int SetSensorLocation (int SensorLocation)

パラメータに、設定したいプロパティ値を指定してください。

未設定時はコマンドを送信しません。

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

取得方法

int GetSensorLocation ()

戻り値として、設定されているセンサー設定(前方/後方)を返します。

未設定時は、デフォルトとして CLS_PROPERTY_DEFAULT(999999) を返します。

使用例

```
int location;
```

```
printer.SetSensorLocation(LabelConst.CLS_SENS_LOCATION_ADJUSTABLE);  
location = printer.GetSensorLocation();
```

2.2.35. CommandInterpreterInAction プロパティ

形式

int CommandInterpreterInAction

属性

Read

説明

このプロパティは、プリンターがコマンド処理中であることを示します。
 このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetCommandInterpreterInAction ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.2.36. PaperError プロパティ

形式

int PaperError

属性

Read

説明

このプロパティは、プリンターのペーパーエラー状態を示します。
 このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetPaperError ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.2.37. RibbonEnd プロパティ

形式

int RibbonEnd

属性

Read

説明

このプロパティは、プリンターのリボンエンド状態を示します。

このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetRibbonEnd ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.2.38. BatchProcessing プロパティ

形式

int BatchProcessing

属性

Read

説明

このプロパティは、プリンターがバッチ処理印字中であることを示します。
 このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetBatchProcessing ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.2.39. Printing プロパティ

形式

int Printing

属性

Read

説明

このプロパティは、プリンターが印字中であることを示します。
 このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetPrinting ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.2.40. Pause プロパティ

形式

int Pause

属性

Read

説明

このプロパティは、プリンターがポーズ中であることを示します。

このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetPause ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.2.41. WaitingForPeeling プロパティ

形式

int WaitingForPeeling

属性

Read

説明

このプロパティは、プリンターが剥離待ち中であることを示します。
 このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetWaitingForPeeling ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.2.42. PrintHeadLowTemp プロパティ

形式

int PrintHeadLowTemp

属性

Read

説明

このプロパティは、プリンターがプリントヘッド低温エラーであることを示します。

このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetPrintHeadLowTemp ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.2.43. PrintHeadFailure プロパティ

形式

int PrintHeadFailure

属性

Read

説明

このプロパティは、プリンターがプリントヘッド切れエラーであることを示します。
 このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetPrintHeadFailure ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.2.44. PrintHeadOverheat プロパティ

形式

int PrintHeadOverheat

属性

Read

説明

このプロパティは、プリンターがプリントヘッドオーバーヒートエラーであることを示します。
このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetPrintHeadOverheat ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.2.45. MechanismOpen プロパティ

形式

int MechanismOpen

属性

Read

説明

このプロパティは、プリンターのプリントヘッドオープンエラーであることを示します。

このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetMechanismOpen ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.2.46. AutoCutterError プロパティ

形式

int AutoCutterError

属性

Read

説明

このプロパティは、プリンターのオートカッター異常を示します。
 このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetAutoCutterError ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.2.47. FanMotorError プロパティ

形式

int FanMotorError

属性

Read

説明

このプロパティは、プリンターのファンモーターエラーを示します。
 このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetFanMotorError ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.2.48. MiscError プロパティ

形式

int MiscError

属性

Read

説明

このプロパティは、本 SDK がサポートしているステータスプロパティ以外のエラーが、プリンターで発生していることを示します。

このプロパティを参照する際は、あらかじめ [PrinterCheck メソッド](#)により、プリンターからステータスを取得しておく必要があります。

設定方法

なし。

取得方法

int GetMiscError ()

戻り値として、以下の値を返します。

値	意味
0	No
1	Yes
CLS_PROPERTY_DEFAULT(999999)	ステータス未取得

使用例

[PrinterCheck メソッド](#)を参照

2.3. LabelDesign クラス

2.3.1. コンストラクタ

形式

LabelDesign ()

パラメータ

ありません。

説明

Windows ライブラリのコンストラクタです。インスタンスを生成します。

戻り値

ありません。

使用例

```
LabelDesign design = new LabelDesign();
```


2.3.2. DrawTextPtrFont メソッド

形式

- 1) int DrawTextPtrFont (String data, int locale,
int font, int rotation, int hexp, int vexp, int size, int x, int y)
- 2) int DrawTextPtrFont (String data, int locale,
int font, int rotation, int hexp, int vexp, int size, int x, int y, int encoding)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
data	[IN]	印字文字列	※文字コード変換できない文字は、空白に置き換える。
locale	[IN]	ロケール(定数)	CLS_LOCALE_JP: 日本モデル用ロケール CLS_LOCALE_OTHER: 海外モデル用ロケール CLS_LOCALE_CN: 中国モデル用ロケール CLS_LOCALE_KR: 韓国モデル用ロケール
font	[IN]	文字種(定数)	CLS_PRT_FNT_0: システムフォント 0 CLS_PRT_FNT_1: システムフォント 1 CLS_PRT_FNT_2: システムフォント 2 CLS_PRT_FNT_3: システムフォント 3 CLS_PRT_FNT_4: システムフォント 4 CLS_PRT_FNT_5: システムフォント 5 CLS_PRT_FNT_6: システムフォント 6 CLS_PRT_FNT_7: システムフォント 7 CLS_PRT_FNT_8: システムフォント 8 CLS_PRT_FNT_TRIUMVIRATE: スムースフォント CLS_PRT_FNT_TRIUMVIRATE_B: スムースフォント(Bold) CLS_PRT_FNT_KANJI: 漢字(横書き) CLS_PRT_FNT_KANJIT: 漢字(縦書き)
rotation	[IN]	回転方向(定数)	CLS_RT_NORMAL: 回転なし CLS_RT_RIGHT90: 右 90 度回転 CLS_RT_ROTATE180: 右 180 度回転 CLS_RT_LEFT90: 左 90 度回転
hexp	[IN]	水平拡大率	1～24
vexp	[IN]	垂直拡大率	1～24
size	[IN]	文字サイズ(定数)	CLS_PRT_FNT_SIZE_4: 文字種ポイント数(4pt) CLS_PRT_FNT_SIZE_5: 文字種ポイント数(5pt) CLS_PRT_FNT_SIZE_6: 文字種ポイント数(6pt) CLS_PRT_FNT_SIZE_8: 文字種ポイント数(8pt) CLS_PRT_FNT_SIZE_10: 文字種ポイント数(10pt) CLS_PRT_FNT_SIZE_12: 文字種ポイント数(12pt) CLS_PRT_FNT_SIZE_14: 文字種ポイント数(14pt) CLS_PRT_FNT_SIZE_18: 文字種ポイント数(18pt) CLS_PRT_FNT_SIZE_24: 文字種ポイント数(24pt) CLS_PRT_FNT_SIZE_30: 文字種ポイント数(30pt) CLS_PRT_FNT_SIZE_36: 文字種ポイント数(36pt) CLS_PRT_FNT_SIZE_48: 文字種ポイント数(48pt) CLS_PRT_FNT_KANJI_SIZE_16: 漢字文字種(16ドット) CLS_PRT_FNT_KANJI_SIZE_24: 漢字文字種(24ドット) CLS_PRT_FNT_KANJI_SIZE_32: 漢字文字種(32ドット) ※ CLS_PRT_FNT_KANJI_SIZE_48: 漢字文字種(48ドット) ※ ※中国モデル用ロケールでは、非サポート。
x	[IN]	印字位置(X 座標)	0000～9999

y	[IN]	印字位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。
encoding	[IN]	印字文字列の文字コード (定数)	「 3.3.パラメータ 」の No9 を参照

説明

プリンターのシステムフォントを用いて、回転・縦横拡大率・フォント種・印字位置等・指定条件で、入力された内容の文字を描画します。

文字サイズは、文字種ごとに以下のパターンを指定可能です。

漢字: 16,24,32,48 ※ドット単位

スムースフォント: 4,5,6,8,10,12,14,18,24,30,36,48

上記以外: 文字種ごとにサイズ固定

文字コードが省略された場合は、自動的にロケールに合わせた文字コードが設定されます。文字コードに UTF-8(CLS_ENC_CDPG_UTF_8)が指定できるモデルは CL-S700III のみです。(2024 年 4 月現在)

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
// Locale:OTHER
design.DrawTextPtrFont("DrawTextPtrFont",
    LabelConst.CLS_LOCALE_OTHER, LabelConst.CLS_PRT_FNT_TRIUMVIRATE,
    LabelConst.CLS_RT_NORMAL, 1, 1, LabelConst.CLS_PRT_FNT_SIZE_10,
    100, 100);

// Locale:JP
design.DrawTextPtrFont("テキスト印刷(プリンターフォント)",
    LabelConst.CLS_LOCALE_JP, LabelConst.CLS_PRT_FNT_KANJI,
    LabelConst.CLS_RT_NORMAL, 1, 1,
    LabelConst.CLS_PRT_FNT_KANJI_SIZE_16, 100, 300);

// Locale:CN
design.DrawTextPtrFont("测试打印", LabelConst.CLS_LOCALE_CN,
    LabelConst.CLS_PRT_FNT_KANJI, LabelConst.CLS_RT_NORMAL, 1, 1,
    LabelConst.CLS_PRT_FNT_KANJI_SIZE_16, 100, 300);

// Locale:KR
design.DrawTextPtrFont("테스트 인쇄", LabelConst.CLS_LOCALE_KR,
    LabelConst.CLS_PRT_FNT_KANJI, LabelConst.CLS_RT_NORMAL, 1, 1,
    LabelConst.CLS_PRT_FNT_KANJI_SIZE_16, 100, 300);
```

2.3.3. DrawTextDLFont メソッド

形式

```
int DrawTextDLFont (String data, int encoding,  
                    String fontID, int rotation, int hexp, int vexp, int point, int x, int y)
```

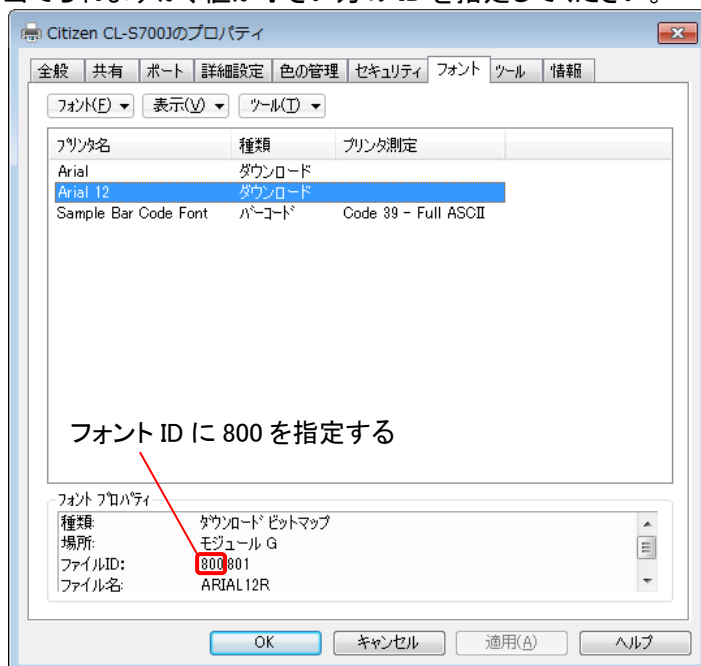
パラメータ

値	[IN/OUT]	意味	設定可能範囲
data	[IN]	印字文字列	※文字コード変換できない文字は、空白に置き換える。
encoding	[IN]	印字文字列の文字コード(定数)	「 3.3.パラメータ 」の No9 を参照
fontID	[IN]	プリンターにダウンロードしたフォントの ID	<ul style="list-style-type: none"> ・fontID が全て数字の場合 ビットマップダウンロードフォント ・fontID に数字以外が含まれる場合 TrueType ダウンロードフォント
rotation	[IN]	回転方向(定数)	CLS_RT_NORMAL: 回転なし CLS_RT_RIGHT90: 右 90 度回転 CLS_RT_ROTATE180: 右 180 度回転 CLS_RT_LEFT90: 左 90 度回転
hexp	[IN]	水平拡大率	1～24
vexp	[IN]	垂直拡大率	1～24
point	[IN]	文字サイズ	001～999
x	[IN]	印字位置(X 座標)	0000～9999
y	[IN]	印字位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。

説明

プリンターにダウンロードしたフォントを使用して、回転・縦横拡大率・印字位置・文字サイズ等・指定条件で、入力された内容の文字を描画します。

ビットマップダウンロードフォントにつきましては、フォントをプリンターへダウンロードした際に 2 つの ID が割り当てられますが、値が小さい方の ID を指定してください。



文字コードに UTF-8 (CLS_ENC_CDPG_UTF_8) が指定できるモデルは CL-S700III のみです。(2022 年 10 月現在)

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
// TrueTypeダウンロードフォント
design.DrawTextDLFont("TrueType",
    LabelConst.CLS_ENC_CDPG_IBM850, "S50",
    LabelConst.CLS_RT_NORMAL, 1, 1, 12, 100, 100);

// ビットマップダウンロードフォント
design.DrawTextDLFont("Bitmap",
    LabelConst.CLS_ENC_CDPG_IBM850, "800",
    LabelConst.CLS_RT_NORMAL, 1, 1, 12, 100, 200);
```

2.3.4. DrawTextPCFont メソッド

形式

- 1) int DrawTextPCFont (String data, String fontName, int rotation, int hRatio, int vRatio, int point, int style, int x, int y)
- 2) int DrawTextPCFont (String data, String fontName, int rotation, int hRatio, int vRatio, int point, int style, int x, int y, int resolution)
- 3) int DrawTextPCFont (String data, String fontName, int rotation, int hRatio, int vRatio, int point, int style, int x, int y, int resolution, int measurementUnit)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
data	[IN]	印字文字列	
fontName	[IN]	印字文字列のフォント名(名称)	フォント名称
rotation	[IN]	回転方向	CLS_RT_NORMAL: 回転なし CLS_RT_RIGHT90: 右 90 度回転 CLS_RT_ROTATE180: 右 180 度回転 CLS_RT_LEFT90: 左 90 度回転
hRatio	[IN]	水平拡大率	1～1000(%単位)
vRatio	[IN]	垂直拡大率	1～1000(%単位)
point	[IN]	文字サイズ	
style	[IN]	文字スタイル(定数)	CLS_FNT_DEFAULT: 無し CLS_FNT_BOLD: 太字 CLS_FNT_REVERSE: 反転 CLS_FNT_UNDERLINE: アンダーライン CLS_FNT_ITALIC: イタリック CLS_FNT_STRIKEOUT: 取り消し線 ※複数指定時は「 」で連結する。
x	[IN]	印字位置(X 座標)	0000～9999
y	[IN]	印字位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。
resolution	[IN]	解像度(dpi)	CLS_PRT_RES_203(203dpi) CLS_PRT_RES_300(300dpi) ※省略時は、CLS_PRT_RES_203 となる。
measurementUnit	[IN]	プリンターの単位選択設定	CLS_UNIT_MILLI CLS_UNIT_INCH ※省略時は、CLS_UNIT_INCH となる。

説明

文字コード・回転・縦横拡大率・文字スタイル・文字サイズ・印字位置・解像度等・指定条件で、入力された内容の文字を描画します。

本メソッドは内部的に、入力された内容を画像化し、画像化したデータをプリンターへ登録してから印刷を行っていますが、このとき、プリンターのメモリ容量の問題等で、印字画像をプリンターへ登録可能であるか・登録されたかのチェックは行いません。

指定した文字サイズで印刷を行うには、解像度をプリンターに合わせて指定してください。

イメージの回転指定をする場合は、基準位置補正のため解像度と単位選択設定をプリンターに合わせて指定する必要があります。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
// 解像度指定 省略(203dpi)、単位指定 省略(inch)
design.DrawTextPCFont("DrawTextPCFont", "Arial",
    LabelConst.CLS_RT_NORMAL, 100, 100, 10,
    (LabelConst.CLS_FNT_BOLD | LabelConst.CLS_FNT_ITALIC), 10, 0);

// 解像度指定 300dpi、単位指定 省略(inch)
design.DrawTextPCFont("DrawTextPCFont", "Tahoma",
    LabelConst.CLS_RT_NORMAL, 100, 100, 10,
    LabelConst.CLS_FNT_DEFAULT, 10, 50, LabelConst.CLS_PRT_RES_300);

// 解像度指定 300dpi、単位指定 mm
design.DrawTextPCFont("DrawTextPCFont", "Tahoma",
    LabelConst.CLS_RT_RIGHT90, 100, 100, 10,
    LabelConst.CLS_FNT_DEFAULT, 10, 50,
    LabelConst.CLS_PRT_RES_300, LabelConst.CLS_UNIT_MILLI);
```

2.3.5. DrawNVBitmap メソッド

形式

int DrawNVBitmap (String name, int hexp, int vexp, int x, int y)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
name	[IN]	画像ファイル名	以下の条件を除いた ASCII コードが使用可能 ・先頭がアンダースコアは不可 ・以下の文字は使用不可(禁則文字) ¥ / : * ? " < >
hexp	[IN]	水平拡大率	1 ~ 24
vexp	[IN]	垂直拡大率	1 ~ 24
x	[IN]	印字位置(X 座標)	0000 ~ 9999
y	[IN]	印字位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。

説明

縦横拡大率・印字位置等・指定条件で、プリンターに登録されている画像データを描画します。
指定された画像ファイルが、プリンターに登録されているかのチェックは行いません。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

[StoreNVBitmap メソッド](#)を参照

2.3.6. DrawBitmap メソッド

形式

- 1) int DrawBitmap (String filePath, int rotation, int width, int height, int x, int y)
- 2) int DrawBitmap (String filePath, int rotation, int width, int height, int x, int y, int resolution, int measurementUnit)
- 3) int DrawBitmap (String filePath, int rotation, int width, int height, int x, int y, int resolution, int measurementUnit, int mode)
- 4) int DrawBitmap (Bitmap bitmap, int rotation, int width, int height, int x, int y)
- 5) int DrawBitmap (Bitmap bitmap, int rotation, int width, int height, int x, int y, int resolution, int measurementUnit)
- 6) int DrawBitmap (Bitmap bitmap, int rotation, int width, int height, int x, int y, int resolution, int measurementUnit, int mode)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
filePath	[IN]	対象のファイルパス	
bitmap	[IN]	ビットマップ型データ	
rotation	[IN]	回転方向(定数)	CLS_RT_NORMAL: 回転なし CLS_RT_RIGHT90: 右 90 度回転 CLS_RT_ROTATE180: 右 180 度回転 CLS_RT_LEFT90: 左 90 度回転
width	[IN]	横サイズ(ピクセル単位)	
height	[IN]	縦サイズ(ピクセル単位)	
x	[IN]	印字位置(X 座標)	0000～9999
y	[IN]	印字位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。
resolution	[IN]	解像度(dpi)	CLS_PRT_RES_203(203dpi) CLS_PRT_RES_300(300dpi) ※省略時は、CLS_PRT_RES_203 となる。
measurementUnit	[IN]	プリンターの単位選択設定	CLS_UNIT_MILLI CLS_UNIT_INCH ※省略時は、CLS_UNIT_INCH となる。
mode	[IN]	ビットマップモード	CLS_BM_MODE_HT_THRESHOLD: ハーフトーン しきい値 CLS_BM_MODE_HT_DITHER: ハーフトーン ディザ

説明

このメソッドは、ビットマップのファイル名またはビットマップ型データを回転方向・縦横サイズ・印字位置等の条件を指定して描画します。

本メソッドは内部的に、指定された画像をプリンターへ登録してから描画を行っていますが、このとき、プリンターのメモリ容量の問題等で、画像をプリンターへ登録可能であるか・登録されたかのチェックは行いません。イメージの回転指定をする場合は、基準位置補正のため解像度と単位選択設定をプリンターに合わせて指定する必要があります。

描画可能なビットマップ形式は、BMP/GIF/EXIF/JPG/PNG/TIFF です。

描画される画像サイズは、指定された縦横サイズに合わせて、アスペクト比を維持した状態でリサイズします。

例: 以下の場合、リサイズ後の画像サイズは「200x50」ピクセルとなる。

画像サイズ: 400x100 ピクセル

横サイズ: 200

縦サイズ:200

指定のサイズが0の場合、対象の画像サイズを基準とする。

例: 以下の場合、リサイズ後の画像サイズは「800x200」ピクセルとなる。

画像サイズ:400x100 ピクセル

横サイズ:0

縦サイズ:200

モードは、ハーフトーン処理方法を指定します。モードが省略された場合は、CLS_BM_MODE_HT_DITHER で印刷します。モードに関する詳細は、次の通りです。

ハーフトーン ハーフトーン処理方法を指定します。

設定値	説明
CLS_BM_MODE_HT_THRESHOLD	しきい値 文字の印刷に適します。
CLS_BM_MODE_HT_DITHER	ディザー グラフィックの印刷に適します。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
// 解像度指定 省略(203dpi)、単位指定 省略(inch)
design.DrawBitmap("C:¥¥TestFolder¥¥Sample.bmp",
    LabelConst.CLS_RT_NORMAL, 0, 0, 10, 10);

// 解像度指定 300dpi、単位指定 mm
design.DrawBitmap("C:¥¥TestFolder¥¥Sample.bmp",
    LabelConst.CLS_RT_RIGHT90, 0, 0, 10, 10,
    LabelConst.CLS_PRT_RES_300, LabelConst.CLS_UNIT_MILLI);
```

2.3.7. DrawBarCode メソッド

形式

int DrawBarCode (String data, int symbology,
int rotation, int thick, int narrow, int height, int x, int y, int showText)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
data	[IN]	バーコードデータ文字列	ASCII コード文字列
symbology	[IN]	バーコードの種類(定数)	CLS_BCS_CODE39: Code 3 of 9 CLS_BCS_UPCA: UPC-A CLS_BCS_UPCE: UPC-E CLS_BCS_INTERLEAVED25: Interleaved 2 of 5 CLS_BCS_CODE128: Code 128 CLS_BCS_EAN13: EAN-13(JAN-13) CLS_BCS_EAN8: EAN-8(JAN-8) CLS_BCS_HIBC: HIBC CLS_BCS_CODABAR: CODABAR(NW-7) CLS_BCS_INT25: Int 2 of 5 CLS_BCS_PLESSEY: Plessey CLS_BCS_CASECODE: CASE CODE CLS_BCS_UPC2DIG: UPC 2DIG ADD CLS_BCS_UPC5DIG: UPC 5DIG ADD CLS_BCS_CODE93: Code93 CLS_BCS_ITF14: ITF-14 CLS_BCS_ZIP: ZIP CLS_BCS_ITF16: IFT-16 CLS_BCS_UCCEAN128: UCC/EAN-128 CLS_BCS_INDUSTRIAL25: Industrial 2 of 5 CLS_BCS_UCCEAN128KMART: UCC/EAN-128(for K-MART) CLS_BCS_COOP25: COOP 2 of 5 CLS_BCS_UCCEAN128RANDOMWEIGHT: UCC/EAN-128 Random Weight CLS_BCS_TELEPEN: Telepen
rotation	[IN]	回転方向(定数)	CLS_RT_NORMAL: 回転なし CLS_RT_RIGHT90: 右 90 度回転 CLS_RT_ROTATE180: 右 180 度回転 CLS_RT_LEFT90: 左 90 度回転
thick	[IN]	太バーの幅	1～24
narrow	[IN]	細バーの幅	1～24
height	[IN]	バーコードデータの高さ	001～999
x	[IN]	印字位置(X 座標)	0000～9999
y	[IN]	印字位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。
showText	[IN]	バーコード文字の表示有無(定数)	CLS_BCS_TEXT_HIDE: 非表示 CLS_BCS_TEXT_SHOW: 表示

説明

このメソッドは、一次元のバーコードを描画します。また、バーコードに対して、バーコード種類・太/細バーの幅・バーコードデータの長さ・印字位置・バーコード文字の表示有無等の条件を指定することができます。
※バーコードの種類によって、thick/narrow 比率や、有効なバーコードデータ文字列が異なります。詳細は、

ラベルプリンターのコマンドリファレンス「バーコードフィールドの定義」、「各バーコードの説明」の解説をご確認ください。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
// Code3of9
string code39 = "ABC123456789";
design.DrawBarCode(code39, LabelConst.CLS_BCS_CODE39,
    LabelConst.CLS_RT_NORMAL, 6, 2, 50, 10, 10,
    LabelConst.CLS_BCS_TEXT_SHOW);

// UPC-A
string upca = "01234567890";
design.DrawBarCode(upca, LabelConst.CLS_BCS_UPCA,
    LabelConst.CLS_RT_NORMAL, 2, 2, 50, 10, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

// UPC-E
string upce = "123456";
design.DrawBarCode(upce, LabelConst.CLS_BCS_UPCE,
    LabelConst.CLS_RT_NORMAL, 2, 2, 50, 10, 10,
    LabelConst.CLS_BCS_TEXT_SHOW);

// Interleaved2of5
string interleaved25 = "1234567890";
design.DrawBarCode(interleaved25, LabelConst.CLS_BCS_INTERLEAVED25,
    LabelConst.CLS_RT_NORMAL, 5, 2, 50, 10, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

// Code128
string code128 = "1234567890";
design.DrawBarCode(code128, LabelConst.CLS_BCS_CODE128,
    LabelConst.CLS_RT_NORMAL, 2, 4, 50, 10, 10,
    LabelConst.CLS_BCS_TEXT_SHOW);

// EAN-13
string ean13 = "123456789012";
design.DrawBarCode(ean13, LabelConst.CLS_BCS_EAN13,
    LabelConst.CLS_RT_NORMAL, 3, 3, 50, 10, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

// EAN-8
string ean8 = "1234567";
design.DrawBarCode(ean8, LabelConst.CLS_BCS_EAN8,
    LabelConst.CLS_RT_NORMAL, 4, 4, 50, 10, 10,
    LabelConst.CLS_BCS_TEXT_SHOW);

// HIBC
string hibi = "1234567890";
design.DrawBarCode(hibi, LabelConst.CLS_BCS_HIBC,
    LabelConst.CLS_RT_NORMAL, 6, 2, 50, 10, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

// CODABAR
string codabar = "a1234567890b";
design.DrawBarCode(codabar, LabelConst.CLS_BCS_CODABAR,
    LabelConst.CLS_RT_NORMAL, 6, 2, 40, 10, 10,
```

```
LabelConst.CLS_BCS_TEXT_SHOW);

// Interleaved2of5 W/BARS
string int25 = "1234567890";
design.DrawBarCode(int25, LabelConst.CLS_BCS_INT25,
    LabelConst.CLS_RT_NORMAL, 5, 2, 50, 10, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

// PLESSEY
string plessey = "1234567890";
design.DrawBarCode(plessey, LabelConst.CLS_BCS_PLESSEY,
    LabelConst.CLS_RT_NORMAL, 4, 2, 50, 10, 10,
    LabelConst.CLS_BCS_TEXT_SHOW);

// CASE CODE
string casocode = "1234567890123";
design.DrawBarCode(casocode, LabelConst.CLS_BCS_CASECODE,
    LabelConst.CLS_RT_NORMAL, 5, 2, 50, 10, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

// UPC 2DIG ADD
string upca = "01234567890";
string upc2dig = "12";

design.DrawBarCode(upca, LabelConst.CLS_BCS_UPCA,
    LabelConst.CLS_RT_NORMAL, 2, 2, 50, 10, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

design.DrawBarCode(upc2dig, LabelConst.CLS_BCS_UPC2DIG,
    LabelConst.CLS_RT_NORMAL, 4, 2, 50, 130, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

// UPC 5DIG ADD
string upca = "01234567890";
string upc5dig = "12345";

design.DrawBarCode(upca, LabelConst.CLS_BCS_UPCA,
    LabelConst.CLS_RT_NORMAL, 2, 2, 50, 10, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

design.DrawBarCode(upc5dig, LabelConst.CLS_BCS_UPC5DIG,
    LabelConst.CLS_RT_NORMAL, 4, 2, 50, 130, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

// Code93
string code93 = "1234ABCD";
design.DrawBarCode(code93, LabelConst.CLS_BCS_CODE93,
    LabelConst.CLS_RT_NORMAL, 6, 6, 50, 10, 10,
    LabelConst.CLS_BCS_TEXT_SHOW);

// ITF-14
string itf14 = "1234567890123";
design.DrawBarCode(itf14, LabelConst.CLS_BCS_ITF14,
    LabelConst.CLS_RT_NORMAL, 5, 2, 50, 10, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

// ZIP
string zip = "123456789";
design.DrawBarCode(zip, LabelConst.CLS_BCS_ZIP,
    LabelConst.CLS_RT_NORMAL, 1, 1, 10, 10, 10,
```

```
LabelConst.CLS_BCS_TEXT_SHOW);

// ITF-16
string itf16 = "123456789012345";
design.DrawBarCode(itf16, LabelConst.CLS_BCS_ITF16,
    LabelConst.CLS_RT_NORMAL, 5, 2, 50, 10, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

// UCC/EAN-128
string uccean128 = "1234567890123456789";
design.DrawBarCode(uccean128, LabelConst.CLS_BCS_UCCEAN128,
    LabelConst.CLS_RT_NORMAL, 4, 4, 50, 10, 10,
    LabelConst.CLS_BCS_TEXT_SHOW);

// Industrial2of5
string industrial25 = "1234567890";
design.DrawBarCode(industrial25, LabelConst.CLS_BCS_INDUSTRIAL25,
    LabelConst.CLS_RT_NORMAL, 5, 2, 50, 10, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

// UCC/EAN-128(for K-MART)
string uccean128kmart = "123456789012345678";
design.DrawBarCode(uccean128kmart, LabelConst.CLS_BCS_UCCEAN128KMART,
    LabelConst.CLS_RT_NORMAL, 3, 3, 50, 10, 10,
    LabelConst.CLS_BCS_TEXT_SHOW);

// COOP 2 of 5
string coop25 = "0123456789";
design.DrawBarCode(coop25, LabelConst.CLS_BCS_COOP25,
    LabelConst.CLS_RT_NORMAL, 10, 4, 50, 10, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);

// UCC/EAN-128 Random Weight
string uccean128randomweight = "12345678901234567890123456789099999";
design.DrawBarCode(uccean128randomweight,
    LabelConst.CLS_BCS_UCCEAN128RANDOMWEIGHT,
    LabelConst.CLS_RT_NORMAL, 2, 2, 50, 10, 10,
    LabelConst.CLS_BCS_TEXT_SHOW);

// Telepen
string telepen = "1234567890";
design.DrawBarCode(telepen, LabelConst.CLS_BCS_TELEPEN,
    LabelConst.CLS_RT_NORMAL, 2, 2, 50, 10, 100,
    LabelConst.CLS_BCS_TEXT_SHOW);
```

2.3.8. DrawMaxiCode メソッド

形式

int DrawMaxiCode (String[] data, int rotation, int x, int y)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
data	[IN]	バーコードデータ文字列	ASCII コード文字列
rotation	[IN]	回転方向(定数)	CLS_RT_NORMAL: 回転なし CLS_RT_RIGHT90: 右 90 度回転 CLS_RT_ROTATE180: 右 180 度回転 CLS_RT_LEFT90: 左 90 度回転
x	[IN]	印字位置(X 座標)	0000～9999
y	[IN]	印字位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。

説明

このメソッドは、UPS MaxiCode 形式のバーコードを描画します。また、バーコードに対して、回転・印字位置の条件を指定することができます。

バーコードデータ文字列には、以下 5 つの要素を、①から順にセットしてください。

- ① 5 桁の Zip コード
- ② 4 桁の+4Zip コード
- ③ 3 桁の国別コード
- ④ 3 桁の class of service code
- ⑤ 84 桁以内のデータ文字列

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
string[] data = new string[] { "90501", "6282", "840", "001", "1Z00004951" };
design.DrawMaxiCode(data, LabelConst.CLS_RT_NORMAL, 100, 0);
```

2.3.9. DrawPDF417 メソッド

形式

int DrawPDF417 (String data, int encoding, int rotation, int exp, int ECLevel, int x, int y)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
data	[IN]	バーコードデータ文字列	※指定の文字コードで使用可能な文字
encoding	[IN]	バーコードデータの文字コード(定数)	「 3.3.パラメータ 」の No9 を参照
rotation	[IN]	回転方向(定数)	CLS_RT_NORMAL: 回転なし CLS_RT_RIGHT90: 右 90 度回転 CLS_RT_ROTATE180: 右 180 度回転 CLS_RT_LEFT90: 左 90 度回転
exp	[IN]	拡大率	1～5
ECLevel	[IN]	エラー修正レベル(定数)	CLS_PDF417_EC_LEVEL_0: レベル 0 CLS_PDF417_EC_LEVEL_1: レベル 1 CLS_PDF417_EC_LEVEL_2: レベル 2 CLS_PDF417_EC_LEVEL_3: レベル 3 CLS_PDF417_EC_LEVEL_4: レベル 4 CLS_PDF417_EC_LEVEL_5: レベル 5 CLS_PDF417_EC_LEVEL_6: レベル 6 CLS_PDF417_EC_LEVEL_7: レベル 7 CLS_PDF417_EC_LEVEL_8: レベル 8
x	[IN]	印字位置(X 座標)	0000～9999 ※x,y は、左下を基準(0,0)として位置とする。
y	[IN]	印字位置(Y 座標)	

説明

このメソッドは、PDF-417 形式のバーコードを描画します。また、バーコードに対して、エンコード・回転・拡大率・エラー修正レベル・印字位置の条件を指定することができます。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
design.DrawPDF417("0123456789",
    LabelConst.CLS_ENC_CDPG_IBM850,
    LabelConst.CLS_RT_NORMAL, 3,
    LabelConst.CLS_PDF417_EC_LEVEL_0, 0, 0);
```

2.3.10. DrawDataMatrix メソッド

形式

int DrawDataMatrix (String data, int encoding, int rotation, int exp, int ECLevel, int x, int y)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
data	[IN]	バーコードデータ文字列	・数字のみの場合、3116 文字以内 ・英数字混在の場合、2335 文字以内 ※指定の文字コードで使用可能な文字
encoding	[IN]	バーコードデータの文字コード(定数)	「 3.3.パラメータ 」の No9 を参照
rotation	[IN]	回転方向(定数)	CLS_RT_NORMAL: 回転なし CLS_RT_RIGHT90: 右 90 度回転 CLS_RT_ROTATE180: 右 180 度回転 CLS_RT_LEFT90: 左 90 度回転
exp	[IN]	拡大率	1～15
ECLevel	[IN]	エラー修正レベル(定数)	CLS_DATAMATRIX_EC_LEVEL_200: 200
x	[IN]	印字位置(X 座標)	0000～9999
y	[IN]	印字位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。

説明

このメソッドは、DataMatrix 形式のバーコードを描画します。また、バーコードに対して、エンコード・回転・拡大率・印字位置の条件を指定することができます。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
design.DrawDataMatrix("0123456789",
    LabelConst.CLS_ENC_CDPG_IBM850,
    LabelConst.CLS_RT_NORMAL, 15,
    LabelConst.CLS_DATAMATRIX_EC_LEVEL_200, 0, 0);
```


2.3.11. DrawQRCode メソッド

形式

int DrawQRCode (String data, int encoding, int rotation, int exp, int ECLevel, int x, int y)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
data	[IN]	バーコードデータ文字列	※指定の文字コードで使用可能な文字
encoding	[IN]	バーコードデータの文字コード(定数)	「 3.3.パラメータ 」の No9 を参照
rotation	[IN]	回転方向(定数)	CLS_RT_NORMAL: 回転なし CLS_RT_RIGHT90: 右 90 度回転 CLS_RT_ROTATE180: 右 180 度回転 CLS_RT_LEFT90: 左 90 度回転
exp	[IN]	拡大率	1～15
ECLevel	[IN]	エラー修正レベル(定数)	CLS_QRCODE_EC_LEVEL_L: レベル L(7%) CLS_QRCODE_EC_LEVEL_M: レベル M(15%) CLS_QRCODE_EC_LEVEL_Q: レベル Q(25%) CLS_QRCODE_EC_LEVEL_H: レベル H(30%)
x	[IN]	印字位置(X 座標)	0000～9999
y	[IN]	印字位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。

説明

このメソッドは、QR コードを描画します。また、QR コードに対して、エンコード・回転・拡大率・エラー修正レベル・印字位置の条件を指定することができます。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
design.DrawQRCode("アイエオ12345",
    LabelConst.CLS_ENC_CDPG_SHIFT_JIS,
    LabelConst.CLS_RT_NORMAL, 10,
    LabelConst.CLS_QRCODE_EC_LEVEL_H, 100, 100);
```

2.3.12. DrawAztec メソッド

形式

int DrawAztec (String data, int encoding, int rotation, int exp, int ECLevel, int x, int y)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
data	[IN]	バーコードデータ文字列	※指定の文字コードで使用可能な文字
encoding	[IN]	バーコードデータの文字コード(定数)	「 3.3.パラメータ 」の No9 を参照
rotation	[IN]	回転方向(定数)	CLS_RT_NORMAL: 回転なし CLS_RT_RIGHT90: 右 90 度回転 CLS_RT_ROTATE180: 右 180 度回転 CLS_RT_LEFT90: 左 90 度回転
exp	[IN]	拡大率	1～15
ECLevel	[IN]	エラー修正レベル(定数)	CLS_AXTEC_EC_LEVEL_000: レベル 0(誤り訂正率 23%)
x	[IN]	印字位置(X 座標)	0000～9999
y	[IN]	印字位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。

説明

このメソッドは、Aztec 形式のバーコードを描画します。また、バーコードに対して、エンコード・回転・拡大率・印字位置の条件を指定することができます。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
design.DrawAztec("0123456789",
    LabelConst.CLS_ENC_CDPG_IBM850,
    LabelConst.CLS_RT_NORMAL, 10,
    LabelConst.CLS_AXTEC_EC_LEVEL_000, 0, 0);
```

2.3.13. DrawGS1DataBar メソッド

形式

int DrawGS1DataBar (String[] data, int type, int rotation, int exp, int x, int y)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
data	[IN]	バーコードデータ文字列	ASCII コード文字列
encoding	[IN]	バーコードタイプ(定数)	CLS_GS1_DATABAR_OMNI_DIRECTIONAL: Omni-directional CLS_GS1_DATABAR_COMPOSITE: Composite CLS_GS1_DATABAR_TRUNCATION: Truncation CLS_GS1_DATABAR_STACKED: Stacked CLS_GS1_DATABAR_STACKED_OMNI_DIRECTIONAL: Stacked Omni-directional CLS_GS1_DATABAR_LIMITED: Limited CLS_GS1_DATABAR_EXPANDED: Expanded
rotation	[IN]	回転方向(定数)	CLS_RT_NORMAL: 回転なし CLS_RT_RIGHT90: 右 90 度回転 CLS_RT_ROTATE180: 右 180 度回転 CLS_RT_LEFT90: 左 90 度回転
exp	[IN]	拡大率	1～15
x	[IN]	印字位置(X 座標)	0000～9999
y	[IN]	印字位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。

説明

このメソッドは、GS1 DataBar 形式のバーコードを描画します。また、バーコードに対して、バーコードタイプ・回転・拡大率・印字位置等の条件を指定することができます。

※バーコードの種類によって、指定可能なバーコードデータ文字列などが異なります。詳細は、ラベルプリンターのコマンドリファレンス「バーコード W1k(国内モデル、海外モデル共通): GS1 DataBar (RSS)」の解説をご確認ください。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
// GS1 DataBar Omni-Directional
string[] omnidirectional = new string[] { "1234567890123" };
design.DrawGS1DataBar(omnidirectional,
    LabelConst.CLS_GS1_DATABAR_OMNI_DIRECTIONAL,
    LabelConst.CLS_RT_NORMAL, 3, 10, 10);

// GS1 DataBar Composite
string[] composit = new string[] {
    "1234567890123", "1234567890-07/07/07" };
design.DrawGS1DataBar(composit,
    LabelConst.CLS_GS1_DATABAR_COMPOSITE,
    LabelConst.CLS_RT_NORMAL, 3, 10, 10);

// GS1 DataBar Truncation
string[] truncation = new string[] { "1234567890123" };
design.DrawGS1DataBar(truncation,
    LabelConst.CLS_GS1_DATABAR_TRUNCATION,
    LabelConst.CLS_RT_NORMAL, 3, 10, 10);
```

```
// GS1 DataBar Stacked
string[] stacked = new string[] { "1234567890123" };
design.DrawGS1DataBar(stacked,
    LabelConst.CLS_GS1_DATABAR_STACKED,
    LabelConst.CLS_RT_NORMAL, 3, 10, 10);

// GS1 DataBar Stacked-Omni-Directional
string[] stackedOD = new string[] { "1234567890123" };
design.DrawGS1DataBar(stackedOD,
    LabelConst.CLS_GS1_DATABAR_STACKED_OMNI_DIRECTIONAL,
    LabelConst.CLS_RT_NORMAL, 3, 10, 10);

// GS1 DataBar Limited
string[] limited = new string[] { "1234567890123" };
design.DrawGS1DataBar(limited,
    LabelConst.CLS_GS1_DATABAR_LIMITED,
    LabelConst.CLS_RT_NORMAL, 3, 10, 10);

// GS1 DataBar Expanded
string[] expanded = new string[] { "041234567890123" };
design.DrawGS1DataBar(expanded,
    LabelConst.CLS_GS1_DATABAR_EXPANDED,
    LabelConst.CLS_RT_NORMAL, 3, 10, 10);
```

2.3.14. DrawLine メソッド

形式

int DrawLine (int x1, int y1, int x2, int y2, int thickness)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
x1	[IN]	印字開始位置(X 座標、中心点)	0000～9999 ※x,y は、左下を基準(0,0)として位置とする。
y1	[IN]	印字開始位置(Y 座標、中心点)	
x2	[IN]	印字終了位置(X 座標、中心点)	
y2	[IN]	印字終了位置(Y 座標、中心点)	
thickness	[IN]	線幅(中心点を基準)	0000～9999

説明

設定された幅の線を描画します。

印字開始・終了位置は、中心線の座標を示します。線幅を考慮した線の座標が、0～9999 の範囲外になる場合、CLS_E_ILLEGAL(1101)エラーになります。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
// 印字領域が重なる部分を上書きに指定
printer.SetFormatAttribute(1);
```

```
// 縦線
design.DrawLine(20, 30, 20, 300, 10);
```

```
// 横線
design.DrawLine(16, 34, 200, 34, 10);
```

2.3.15. DrawRect メソッド

形式

int DrawRect (int x, int y, int width, int height, int thickness)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
x	[IN]	印字開始位置(X 座標)	0000～9999
y	[IN]	印字開始位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。
width	[IN]	水平幅	0000～9999
height	[IN]	垂直幅	
thickness	[IN]	線幅	0000～9999

説明

設定された寸法の矩形を描画します。

印字開始位置は、線の外周の左下を示します。線幅を太くした際は、印字開始位置から内側に線が太くなります。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
design.DrawRect(20, 30, 180, 280, 10);
```

2.3.16. FillRect メソッド

形式

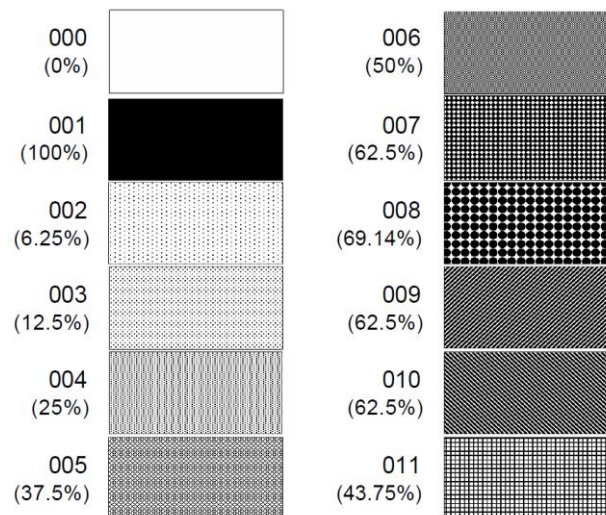
int FillRect (int x, int y, int width, int height, int pattern)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
x	[IN]	印字開始位置(X 座標)	0000～9999 ※x,y は、左下を基準(0,0)として位置とする。
y	[IN]	印字開始位置(Y 座標)	
width	[IN]	水平幅	0000～9999
height	[IN]	垂直幅	
pattern	[IN]	網掛けパターン(定数)	CLS_SHADED_PTN_0: 網掛けパターン 000 CLS_SHADED_PTN_1: 網掛けパターン 001 CLS_SHADED_PTN_2: 網掛けパターン 002 CLS_SHADED_PTN_3: 網掛けパターン 003 CLS_SHADED_PTN_4: 網掛けパターン 004 CLS_SHADED_PTN_5: 網掛けパターン 005 CLS_SHADED_PTN_6: 網掛けパターン 006 CLS_SHADED_PTN_7: 網掛けパターン 007 CLS_SHADED_PTN_8: 網掛けパターン 008 CLS_SHADED_PTN_9: 網掛けパターン 009 CLS_SHADED_PTN_10: 網掛けパターン 010 CLS_SHADED_PTN_11: 網掛けパターン 011

説明

設定された寸法の矩形を描画し、設定されたパターンで内部を網掛けします。



戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
design.FillRect(20, 30, 180, 280, LabelConst.CLS_SHADED_PTN_10);
```

2.3.17. DrawCircle メソッド

形式

int DrawCircle (int x, int y, int radius)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
x	[IN]	印字開始位置(X 座標、中心点)	0000～9999
y	[IN]	印字開始位置(Y 座標、中心点)	※x,y は、左下を基準(0,0)として位置とする。
radius	[IN]	円の半径	0000～0398

説明

設定された中心と半径で円を描画します。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
design.DrawCircle(50, 50, 15);
```


2.3.18. FillCircle メソッド

形式

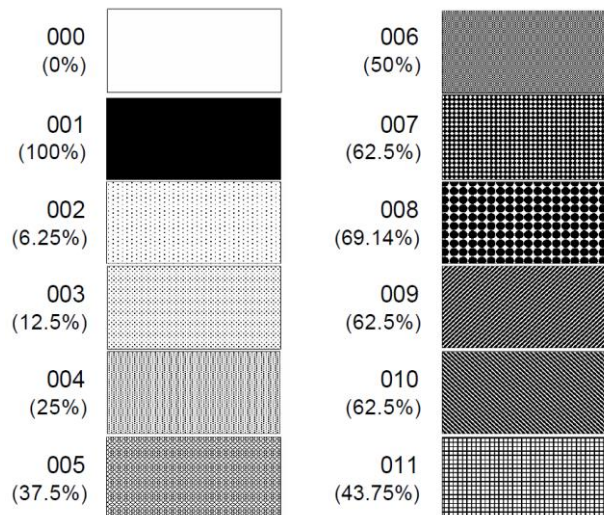
int FillCircle (int x, int y, int radius, int pattern)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
x	[IN]	印字開始位置(X 座標、中心点)	0000～9999
y	[IN]	印字開始位置(Y 座標、中心点)	※x,y は、左下を基準(0,0)として位置とする。
radius	[IN]	円の半径	0000～0398
pattern	[IN]	網掛けパターン(定数)	CLS_SHADED_PTN_0: 網掛けパターン 000 CLS_SHADED_PTN_1: 網掛けパターン 001 CLS_SHADED_PTN_2: 網掛けパターン 002 CLS_SHADED_PTN_3: 網掛けパターン 003 CLS_SHADED_PTN_4: 網掛けパターン 004 CLS_SHADED_PTN_5: 網掛けパターン 005 CLS_SHADED_PTN_6: 網掛けパターン 006 CLS_SHADED_PTN_7: 網掛けパターン 007 CLS_SHADED_PTN_8: 網掛けパターン 008 CLS_SHADED_PTN_9: 網掛けパターン 009 CLS_SHADED_PTN_10: 網掛けパターン 010 CLS_SHADED_PTN_11: 網掛けパターン 011

説明

設定された中心と半径で円を描画し、設定されたパターンで内部を網掛けします。



戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
design.FillCircle(100, 100, 40, LabelConst.CLS_SHADED_PTN_2);
```

2.3.19. DrawPolygon メソッド

形式

```
int DrawPolygon (int[] x, int[] y)
```

パラメータ

値	[IN/OUT]	意味	設定可能範囲
x	[IN]	印字位置(X 座標)	0000～9999
y	[IN]	印字位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。

説明

設定されたポイントで多角形を描画します。多角形の頂点の数だけ、x と y の座標を指定する。x と y の組み合わせは、最低 3 つ必要となり、3 つ未満のときは、CLS_E_ILLEGAL(1101)エラーとなる。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
int[] x = new int[] {100, 200, 250, 150};  
int[] y = new int[] {100, 100, 200, 200};  
  
design.DrawPolygon(x, y);           // 平行四辺形を描画
```

2.3.20. FillPolygon メソッド

形式

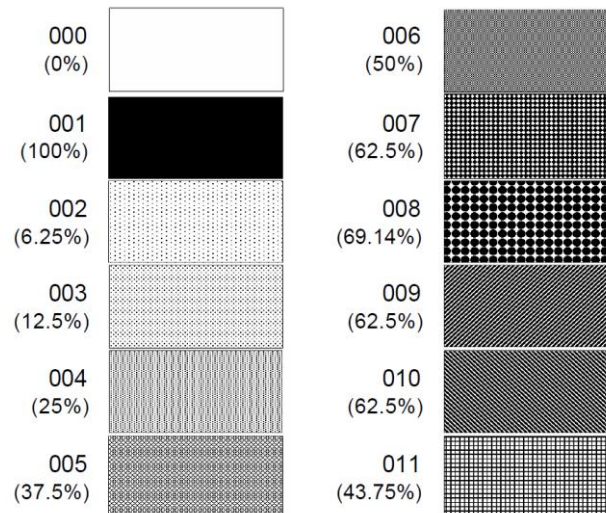
int FillPolygon (int[] x, int[] y, int pattern)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
x	[IN]	印字位置(X 座標)	0000～9999
y	[IN]	印字位置(Y 座標)	※x,y は、左下を基準(0,0)として位置とする。
pattern	[IN]	網掛けパターン(定数)	CLS_SHADED_PTN_0: 網掛けパターン 000 CLS_SHADED_PTN_1: 網掛けパターン 001 CLS_SHADED_PTN_2: 網掛けパターン 002 CLS_SHADED_PTN_3: 網掛けパターン 003 CLS_SHADED_PTN_4: 網掛けパターン 004 CLS_SHADED_PTN_5: 網掛けパターン 005 CLS_SHADED_PTN_6: 網掛けパターン 006 CLS_SHADED_PTN_7: 網掛けパターン 007 CLS_SHADED_PTN_8: 網掛けパターン 008 CLS_SHADED_PTN_9: 網掛けパターン 009 CLS_SHADED_PTN_10: 網掛けパターン 010 CLS_SHADED_PTN_11: 網掛けパターン 011

説明

設定されたポイントで多角形を描画し、設定されたパターンで内部を網掛けします。多角形の頂点の数だけ、x と y の座標を指定する。x と y の組み合わせは、最低 3 つ必要となり、3 つ未満のときは、CLS_E_ILLEGAL(1101)エラーとなる。



戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

```
int[] x = new int[] { 100, 200, 250, 150 };
int[] y = new int[] { 100, 100, 200, 200 };
```

```
design.FillPolygon(x, y, LabelConst.CLS_SHADED_PTN_3); // 平行四辺形を描画
```

2.3.21. EmbedRawDesignCommand メソッド

形式

int EmbedRawDesignCommand (byte[] data)

パラメータ

値	[IN/OUT]	意味	設定可能範囲
data	[IN]	ラベルデザインコマンド	

説明

このメソッドは、ラベルデザインコマンドとして、バイトデータを挿入することができます。
ご使用の際は、他のメソッドに影響を与えない様に注意する必要があります。

戻り値

成功時は CLS_SUCCESS(0) を返します。それ以外のエラーコードは「[2.1 戻り値](#)」を参照してください。

使用例

「DrawRect(100, 200, 300, 250, 100)」と同様のコマンドを、本メソッドを使用して挿入するとき。

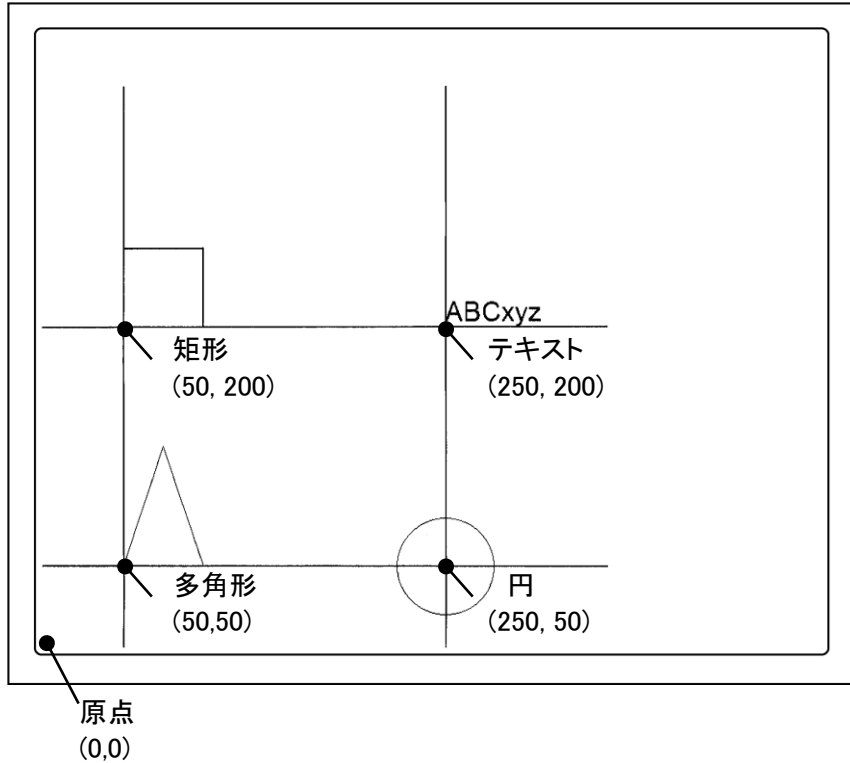
```
byte[] hexdata = new byte[] {
    0x31, 0x58, 0x31, 0x31, 0x30, 0x30, 0x30, 0x30, 0x32, 0x30,
    0x30, 0x30, 0x31, 0x30, 0x30, 0x62, 0x30, 0x33, 0x30, 0x30,
    0x30, 0x32, 0x35, 0x30, 0x30, 0x31, 0x30, 0x30, 0x30, 0x31,
    0x30, 0x30, 0x0D, 0x0A };

design.EmbedRawDesignCommand(hexdata);
```

3. 補足

3.1. 印字位置指定

ラベルにバーコードや文字を印字する場合の位置はラベルの左下が原点となり、原点からの距離を用いて印字位置指定を行います。原点より上方向の距離をY座標、右方向の距離をX座標といいます。



使用例

```
// 多角形
int[] x = new int[] { 50, 75, 100 };
int[] y = new int[] { 50, 125, 50 };
design.DrawPolygon(x, y);

// 矩形
design.DrawRect(50, 200, 50, 50, 1);

// 円
design.DrawCircle(250, 50, 30);

// テキスト
design.DrawTextPCFont("ABCxyz", "Arial",
    LabelConst.CLS_RT_NORMAL, 100, 100, 12,
    LabelConst.CLS_FNT_DEFAULT, 250, 200);

// 罫線
design.DrawLine(0, 200, 350, 200, 1); // 罫線1
design.DrawLine(50, 0, 50, 350, 1); // 罫線2
design.DrawLine(0, 50, 350, 50, 1); // 罫線3
design.DrawLine(250, 0, 250, 350, 1); // 罫線4
```

3.2. ログ機能について

本 SDK は、メソッドの実行やプロパティの読み書きを記録するログ機能をサポートしています。ログ機能を設定する際は、[SetLog メソッド](#)を使用するか、次の書式の設定ファイル「CSJLabelLib.cfg」をライブラリと同じフォルダに配置してください。

＜CSJLabelLib.cfg の例＞

[LogSetting]	…セクション名(固定)
LogMode=1	…ログモードを指定
LogPath=C:\Log	…ログファイルを格納するフォルダを指定
LogMaxSize=10	…ログファイルの最大サイズを MB 単位で指定

設定項目

・ログモード

ログを記録するモードを指定します。

- 0：記録なし
- 1：アクセス履歴の記録
- 2：エラーのみ記録

・格納フォルダ

ログファイルを格納するフォルダを指定します。本設定が指定されていない場合は、カレントフォルダに格納されます。

・ログサイズ

ログファイルの最大容量を MB 単位で指定します。0 を指定した場合は容量制限が解除され可能な限り記録されます。

ログファイル名

ログファイルの拡張子は「.log」です。ファイル名は「CSJLabelLib」の後ろに曜日を表す数字が追加されます。曜日は日曜日を 0、月曜日を 1 として 0 から 6 のいずれかの数字が加えられます。

例) CSJLabelLib_1.log

ログファイルが既に存在し、それが当日以外の場合は、古いファイルを削除してからログを記録します。

ログフォーマット

ログ機能は、メソッド、プロパティの日付、時間、結果のアクセス情報を記録します。

--- メソッドの例1 (Connect) ---

```
2016/03/10 17:14:25.353 4488 009 METHOD call    Connect(0, "192.168.129.130")
2016/03/10 17:14:25.474 4488 009 METHOD result  Connect() -> Success(0)
```

--- メソッドの例2 (Print) ---

```
2016/03/10 17:14:25.474 4488 009 METHOD call    Print([See below], 1)
-----Parameter Detail-----
DrawTextPtrFont("Sample Print", 0, 10, 1, 1, 1, 8, 20, 300) -> 0
DrawQRCode("DrawQRCode", 850, 1, 4, 3, 20, 220) -> 0
FillRect(20, 150, 350, 40, 11) -> 0
DrawBarCode("0123456789", 104, 1, 3, 3, 30, 20, 70, 1) -> 0
-----
2016/03/10 17:14:25.490 4488 009 METHOD result  Print() -> Success(0)
```

--- プロパティの設定例 ---

```
2016/03/10 17:14:25.474 4488 009 PROPERTY get PrintDarkness -> 999999
```

--- プロパティの参照例 ---

```
2016/03/10 17:14:25.474 4488 009 PROPERTY set PrintDarkness <- 10 : Success(0)
```

※ログ機能を使用する場合、全てのメソッドとプロパティのアクセス時にログファイルが更新されますので、SDK の処理が低下してしまうことがあります。

※次のような理由などでファイルの書き込みができない場合はログファイルの記録が行われません。このような場合エラーメッセージなどは表示されませんので、ご注意ください。

- ・書き込み禁止デバイスを指定した場合 (CD-ROM、書き込み禁止のメモリーカードなど)
- ・出力先に十分な領域が残っていない場合
- ・書き込み禁止のログファイルがある場合
- ・ファイルやフォルダのアクセス権がない場合
- ・他のアプリケーションがログファイルを使用している場合

3.3. パラメータ

No	項目	定数名	型	値	説明
1	処理結果	CLS_SUCCESS	int	0	正常終了
		CLS_E_CONNECTED	int	1001	接続済み
		CLS_E_DISCONNECT	int	1002	未接続
		CLS_E_NOTCONNECT	int	1003	接続失敗
		CLS_E_CONNECT_NOTFOUND	int	1004	未対応機種
		CLS_E_USB_BIDIRECTIONAL	int	1010	USB ポートの双方向設定が有効
		CLS_E_ILLEGAL	int	1101	未対応処理または無効パラメータ
		CLS_E_OFFLINE	int	1102	オフライン
		CLS_E_NOEXIST	int	1103	ファイルが存在しない
		CLS_E_FAILURE	int	1104	処理異常
		CLS_E_TIMEOUT	int	1105	書き込みタイムアウト
		CLS_E_NO_LIST	int	1106	プリンターが見つからない
		CLS_EPTR_PAPER_EMPTY	int	1202	用紙切れ
		CLS_EPTR_BADFORMAT	int	1203	画像フォーマット異常
2	プロパティ デフォルト値	CLS_PROPERTY_DEFAULT	int	999999	プロパティのデフォルト値
3	プリンターのステータス	CLS_STS_NO	int	0	ステータス:NO
		CLS_STS_YES	int	1	ステータス:YES
4	接続インターフェース	CLS_PORT_NET	int	0	ネットワーク接続
		CLS_PORT_Bluetooth	int	1	Bluetooth 接続
		CLS_PORT_USB	int	3	USB 接続
		CLS_PORT_COM	int	4	シリアル接続
		CLS_PORT_LPT	int	5	パラレル接続
5	シリアル通信条件	CLS_COM_BAUDRATE_1200	int	1200	ボーレート:1200
		CLS_COM_BAUDRATE_2400	int	2400	ボーレート:2400
		CLS_COM_BAUDRATE_4800	int	4800	ボーレート:4800
		CLS_COM_BAUDRATE_9600	int	9600	ボーレート:9600
		CLS_COM_BAUDRATE_19200	int	19200	ボーレート:19200
		CLS_COM_BAUDRATE_38400	int	38400	ボーレート:38400
		CLS_COM_BAUDRATE_57600	int	57600	ボーレート:57600
		CLS_COM_BAUDRATE_115200	int	115200	ボーレート:115200
		CLS_COM_PARITY_NONE	int	0	パリティ:NONE
		CLS_COM_PARITY_ODD	int	1	パリティ:ODD
		CLS_COM_PARITY_EVEN	int	2	パリティ:EVEN
		CLS_COM_HANDSHAKE_DTRDSR	int	0	フロー制御:DTR/DSR
		CLS_COM_HANDSHAKE_XONXOFF	int	1	フロー制御:XON/XOFF
6	ロケール	CLS_LOCALE_JP	int	0	日本モデル用ロケール
		CLS_LOCALE_OTHER	int	1	海外モデル用ロケール
		CLS_LOCALE_CN	int	2	中国モデル用ロケール
		CLS_LOCALE_KR	int	3	韓国モデル用ロケール
		CLS_LOCALE_UTF_8	int	4	UTF-8 用ロケール
7	文字種	CLS_PRT_FNT_0	int	0	システムフォント:0
		CLS_PRT_FNT_1	int	1	システムフォント:1
		CLS_PRT_FNT_2	int	2	システムフォント:2
		CLS_PRT_FNT_3	int	3	システムフォント:3

		CLS_PRT_FNT_4	int	4	システムフォント:4
		CLS_PRT_FNT_5	int	5	システムフォント:5
		CLS_PRT_FNT_6	int	6	システムフォント:6
		CLS_PRT_FNT_7	int	7	システムフォント:7
		CLS_PRT_FNT_8	int	8	システムフォント:8
		CLS_PRT_FNT_TRIUMVIRATE	int	9	スムースフォント(Triumvirate)
		CLS_PRT_FNT_TRIUMVIRATE_B	int	10	スムースフォント(Triumvirate Bold)
		CLS_PRT_FNT_KANJI	int	11	漢字(横書き)
		CLS_PRT_FNT_KANJIT	int	12	漢字(縦書き)
8	文字サイズ	CLS_PRT_FNT_SIZE_4	int	0	文字種ポイント数(4pt)
		CLS_PRT_FNT_SIZE_5	int	1	文字種ポイント数(5pt)
		CLS_PRT_FNT_SIZE_6	int	2	文字種ポイント数(6pt)
		CLS_PRT_FNT_SIZE_8	int	3	文字種ポイント数(8pt)
		CLS_PRT_FNT_SIZE_10	int	4	文字種ポイント数(10pt)
		CLS_PRT_FNT_SIZE_12	int	5	文字種ポイント数(12pt)
		CLS_PRT_FNT_SIZE_14	int	6	文字種ポイント数(14pt)
		CLS_PRT_FNT_SIZE_18	int	7	文字種ポイント数(18pt)
		CLS_PRT_FNT_SIZE_24	int	8	文字種ポイント数(24pt)
		CLS_PRT_FNT_SIZE_30	int	9	文字種ポイント数(30pt)
		CLS_PRT_FNT_SIZE_36	int	10	文字種ポイント数(36pt)
		CLS_PRT_FNT_SIZE_48	int	11	文字種ポイント数(48pt)
		CLS_PRT_FNT_KANJI_SIZE_16	int	100	漢字文字種(16ドット)
		CLS_PRT_FNT_KANJI_SIZE_24	int	101	漢字文字種(24ドット)
		CLS_PRT_FNT_KANJI_SIZE_32	int	102	漢字文字種(32ドット)
		CLS_PRT_FNT_KANJI_SIZE_48	int	103	漢字文字種(48ドット)
9	エンコード	CLS_ENC_CDPG_DEFAULT	int	0	デフォルト
		CLS_ENC_CDPG_IBM037	int	37	IBM037 IBM EBCDIC (米国-カナダ)
		CLS_ENC_CDPG_IBM437	int	437	IBM437 OEM 米国
		CLS_ENC_CDPG_IBM500	int	500	IBM500 IBM EBCDIC (インターナショナル)
		CLS_ENC_CDPG_IBM737	int	737	ibm737 ギリシャ語 (DOS)
		CLS_ENC_CDPG_IBM775	int	775	ibm775 バルト言語 (DOS)
		CLS_ENC_CDPG_IBM850	int	850	ibm850 西ヨーロッパ語 (DOS)
		CLS_ENC_CDPG_IBM852	int	852	ibm852 中央ヨーロッパ言語 (DOS)
		CLS_ENC_CDPG_IBM855	int	855	IBM855 OEM キリル語
		CLS_ENC_CDPG_IBM857	int	857	ibm857 トルコ語 (DOS)
		CLS_ENC_CDPG_IBM860	int	860	IBM860 ポルトガル語 (DOS)
		CLS_ENC_CDPG_IBM861	int	861	ibm861 アイスランド語 (DOS)
		CLS_ENC_CDPG_IBM863	int	863	IBM863 フランス語 (カナダ)(DOS)
		CLS_ENC_CDPG_IBM865	int	865	IBM865 ノルウェー語 (DOS)
		CLS_ENC_CDPG_CP866	int	866	cp866 キリル語 (DOS)
		CLS_ENC_CDPG_IBM869	int	869	ibm869 ギリシャ語 モダン (DOS)
		CLS_ENC_CDPG_WINDOWS_874	int	874	windows-874 タイ語 (Windows)
		CLS_ENC_CDPG_CP875	int	875	cp875 IBM EBCDIC (ギリシャ語 モダン)
		CLS_ENC_CDPG_SHIFT_JIS	int	932	shift_jis 日本語 (シフト JIS)
		CLS_ENC_CDPG_GB2312	int	936	gb2312 簡体字中国語 (GB2312)
		CLS_ENC_CDPG_KS_C_5601_1987	int	949	ks_c_5601-1987 韓国語

CLS_ENC_CDPG_BIG5	int	950	big5 繁体字中国語 (Big5)
CLS_ENC_CDPG_IBM1026	int	1026	IBM1026 IBM EBCDIC (トルコ語 Latin-5)
CLS_ENC_CDPG_UTF_16	int	1200	utf-16 Unicode
CLS_ENC_CDPG_UNICODEFFFE	int	1201	unicodeFFFE Unicode (Big-Endian)
CLS_ENC_CDPG_WINDOWS_1250	int	1250	windows-1250 中央ヨーロッパ言語 (Windows)
CLS_ENC_CDPG_WINDOWS_1251	int	1251	windows-1251 キリル語 (Windows)
CLS_ENC_CDPG_WINDOWS_1252	int	1252	Windows-1252 西ヨーロッパ言語 (Windows)
CLS_ENC_CDPG_WINDOWS_1253	int	1253	windows-1253 ギリシャ語 (Windows)
CLS_ENC_CDPG_WINDOWS_1254	int	1254	windows-1254 トルコ語 (Windows)
CLS_ENC_CDPG_WINDOWS_1255	int	1255	windows-1255 ヘブライ語 (Windows)
CLS_ENC_CDPG_WINDOWS_1256	int	1256	windows-1256 アラビア語 (Windows)
CLS_ENC_CDPG_WINDOWS_1257	int	1257	windows-1257 バルト語 (Windows)
CLS_ENC_CDPG_WINDOWS_1258	int	1258	windows-1258 ベトナム語 (Windows)
CLS_ENC_CDPG_JOHAB	int	1361	Johab 韓国語 (Johab)
CLS_ENC_CDPG_MACINTOSH	int	10000	macintosh 西ヨーロッパ言語 (Mac)
CLS_ENC_CDPG_X_MAC_JAPANESE	int	10001	x-mac-japanese 日本語 (Mac)
CLS_ENC_CDPG_X_MAC_CHINESETRAD	int	10002	x-mac-chinesetrad 繁体字中国語 (Mac)
CLS_ENC_CDPG_X_MAC_KOREAN	int	10003	x-mac-korean 韓国語 (Mac)
CLS_ENC_CDPG_X_MAC_GREEK	int	10006	x-mac-greek ギリシャ語 (Mac)
CLS_ENC_CDPG_X_MAC_CYRILLIC	int	10007	x-mac-cyrillic キリル語 (Mac)
CLS_ENC_CDPG_X_MAC_CHINESESIMP	int	10008	x-mac-chinesesimp 簡体字中国語 (Mac)
CLS_ENC_CDPG_X_MAC_ROMANIAN	int	10010	x-mac-romanian ルーマニア語 (Mac)
CLS_ENC_CDPG_X_MAC_UKRAINIAN	int	10017	x-mac-ukrainian ウクライナ語 (Mac)
CLS_ENC_CDPG_X_MAC_CE	int	10029	x-mac-ce 中央ヨーロッパ語 (Mac)
CLS_ENC_CDPG_X_MAC_ICELANDIC	int	10079	x-mac-icelandic アイスランド語 (Mac)
CLS_ENC_CDPG_X_MAC_TURKISH	int	10081	x-mac-turkish トルコ語 (Mac)
CLS_ENC_CDPG_X_MAC_CROATIAN	int	10082	x-mac-croatian クロアチア語 (Mac)
CLS_ENC_CDPG_X_CHINESE_CNS	int	20000	x-Chinese-CNS 繁体字中国語 (CNS)
CLS_ENC_CDPG_US_ASCII	int	20127	us-ascii US-ASCII
CLS_ENC_CDPG_X_CP20261	int	20261	x-cp20261 T.61
CLS_ENC_CDPG_IBM290	int	20290	IBM290 IBM EBCDIC (日本語カカナ)
CLS_ENC_CDPG_KOI8_R	int	20866	koi8-r キリル語 (KOI8-R)
CLS_ENC_CDPG_EUC_JP_JIS	int	20932	EUC-JP 日本語 (JIS 0208-1990 and 0212-1990)
CLS_ENC_CDPG_X_CP20936	int	20936	x-cp20936 簡体字中国語 (GB2312-80)
CLS_ENC_CDPG_X_CP20949	int	20949	x-cp20949 韓国語 Wansung
CLS_ENC_CDPG_X_CP21027	int	21027	x-cp21027 Ext Alpha Lowercase
CLS_ENC_CDPG_KOI8_U	int	21866	koi8-u キリル語 (KOI8-R)
CLS_ENC_CDPG_ISO_8859_1	int	28591	iso-8859-1 西ヨーロッパ言語 (ISO)
CLS_ENC_CDPG_ISO_8859_2	int	28592	iso-8859-2 中央ヨーロッパ言語 (ISO)
CLS_ENC_CDPG_ISO_8859_4	int	28594	iso-8859-4 バルト語 (ISO)
CLS_ENC_CDPG_ISO_8859_5	int	28595	iso-8859-5 キリル語 (ISO)
CLS_ENC_CDPG_ISO_8859_7	int	28597	iso-8859-7 ギリシャ語 (ISO)

		CLS_ENC_CDPG_ISO_8859_9	int	28599	iso-8859-9 トルコ語 (ISO)
		CLS_ENC_CDPG_ISO_8859_13	int	28603	iso-8859-13 エストニア語 (ISO)
		CLS_ENC_CDPG_ISO_8859_15	int	28605	iso-8859-15 ラテン語 9 (ISO)
		CLS_ENC_CDPG_ISO_2022_JP	int	50220	iso-2022-jp 日本語 (JIS)
		CLS_ENC_CDPG_CSISO2022JP	int	50221	csISO2022JP 日本語 (JIS-Allow 1 byte Kana)
		CLS_ENC_CDPG_ISO_2022_JP_S	int	50222	iso-2022-jp 日本語 (JIS-Allow 1 byte Kana - SO/SI)
		CLS_ENC_CDPG_ISO_2022_KR	int	50225	iso-2022-kr 韓国語 (ISO)
		CLS_ENC_CDPG_X_CP50227	int	50227	x-cp50227 簡体字中国語 (ISO-2022)
		CLS_ENC_CDPG_EUC_JP	int	51932	euc-jp 日本語 (EUC)
		CLS_ENC_CDPG_EUC_CN	int	51936	EUC-CN 簡体字中国語 (EUC)
		CLS_ENC_CDPG_EUC_KR	int	51949	euc-kr 韓国語 (EUC)
		CLS_ENC_CDPG_HZ_GB_2312	int	52936	hz-gb-2312 簡体字中国語 (HZ)
		CLS_ENC_CDPG_GB18030	int	54936	GB18030 簡体字中国語 (GB18030)
		CLS_ENC_CDPG_UTF_7	int	65000	utf-7 Unicode (UTF-7)
		CLS_ENC_CDPG_UTF_8	int	65001	utf-8 Unicode (UTF-8)
10	文字スタイル	CLS_FNT_DEFAULT	int	0	無し
		CLS_FNT_BOLD	int	8	太字
		CLS_FNT_REVERSE	int	16	反転
		CLS_FNT_UNDERLINE	int	128	アンダーライン
		CLS_FNT_ITALIC	int	256	イタリック
		CLS_FNT_STRIKEOUT	int	512	取り消し線
11	回転方向	CLS_RT_NORMAL	int	1	回転無し(0°)
		CLS_RT_RIGHT90	int	2	右回転(90°)
		CLS_RT_ROTATE180	int	3	反転(180°)
		CLS_RT_LEFT90	int	4	左回転(270°)
12	ビットマップモード	CLS_BM_MODE_HT_THRESHOLD	int	16	ハーフトーン(しきい値)
		CLS_BM_MODE_HT_DITHER	int	32	ハーフトーン(ディザー)
13	バーコードの種類	CLS_BCS_CODE39	int	100	Code 3 of 9
		CLS_BCS_UPCA	int	101	UPC-A
		CLS_BCS_UPCE	int	102	UPC-E
		CLS_BCS_INTERLEAVED25	int	103	Interleaved 2 of 5
		CLS_BCS_CODE128	int	104	Code 128
		CLS_BCS_EAN13	int	105	EAN-13 (JAN-13)
		CLS_BCS_EAN8	int	106	EAN-8 (JAN-8)
		CLS_BCS_HIBC	int	107	HIBC
		CLS_BCS_CODABAR	int	108	CODABAR (NW-7)
		CLS_BCS_INT25	int	109	Int 2 of 5
		CLS_BCS_PLESSEY	int	110	Plessey
		CLS_BCS_CASECODE	int	111	CASE CODE
		CLS_BCS_UPC2DIG	int	112	UPC 2DIG ADD (UPC 用の 2 桁の補足コード)
		CLS_BCS_UPC5DIG	int	113	UPC 5DIG ADD (UPC 用の 5 桁の補足コード)
		CLS_BCS_CODE93	int	114	Code93
		CLS_BCS_ITF14	int	115	(国内モデル)ITF-14

		CLS_BCS_ZIP	int	116	(海外モデル)ZIP
		CLS_BCS_ITF16	int	117	(国内モデル)ITF-16
		CLS_BCS_UCCEAN128	int	118	(海外モデル)UCC/EAN-128
		CLS_BCS_INDUSTRIAL25	int	119	(国内モデル)Industrial 2 of 5
		CLS_BCS_UCCEAN128KMART	int	120	(海外モデル) UCC/EAN-128(for K-MART)
		CLS_BCS_COOP25	int	121	(国内モデル)COOP 2 of 5
		CLS_BCS_UCCEAN128RANDOMWEIGHT	int	122	(海外モデル) UCC/EAN-128 Random Weight
		CLS_BCS_TELEPEN	int	123	Telepen
14	バーコード文字 の表示有無	CLS_BCS_TEXT_HIDE	int	0	非表示
		CLS_BCS_TEXT_SHOW	int	1	表示
15	エラー修正レベ ル(PDF417)	CLS_PDF417_EC_LEVEL_0	int	0	レベル 0
		CLS_PDF417_EC_LEVEL_1	int	1	レベル 1
		CLS_PDF417_EC_LEVEL_2	int	2	レベル 2
		CLS_PDF417_EC_LEVEL_3	int	3	レベル 3
		CLS_PDF417_EC_LEVEL_4	int	4	レベル 4
		CLS_PDF417_EC_LEVEL_5	int	5	レベル 5
		CLS_PDF417_EC_LEVEL_6	int	6	レベル 6
		CLS_PDF417_EC_LEVEL_7	int	7	レベル 7
		CLS_PDF417_EC_LEVEL_8	int	8	レベル 8
16	エラー修正レベ ル(Data Matrix)	CLS_DATAMATRIX_EC_LEVEL_200	int	200	
17	エラー修正レベ ル(QR Code)	CLS_QRCODE_EC_LEVEL_L	int	0	エラー修正レベル L(7%)
		CLS_QRCODE_EC_LEVEL_M	int	1	エラー修正レベル M(15%)
		CLS_QRCODE_EC_LEVEL_Q	int	2	エラー修正レベル Q(25%)
		CLS_QRCODE_EC_LEVEL_H	int	3	エラー修正レベル H(30%)
18	エラー修正レベ ル(Aztec)	CLS_AXTEC_EC_LEVEL_000	int	0	誤り訂正率 23%
19	バーコードタイ プ(GS1 DataBar)	CLS_GS1_DATABAR_OMNI_DIRECTIONAL	int	0	GS1DataBar Omni-directional
		CLS_GS1_DATABAR_COMPOSITE	int	1	GS1DataBar Composite
		CLS_GS1_DATABAR_TRUNCATION	int	2	GS1DataBar Truncation
		CLS_GS1_DATABAR_STACKED	int	3	GS1DataBar Stacked
		CLS_GS1_DATABAR_STACKED_OMNI_DIRECTIONAL	int	4	GS1DataBar Stacked Omni-directional
		CLS_GS1_DATABAR_LIMITED	int	5	GS1DataBar Limited
		CLS_GS1_DATABAR_EXPANDED	int	6	GS1DataBar Expanded
20	網掛けパターン	CLS_SHADED_PTN_0	int	0	網掛けパターン:000
		CLS_SHADED_PTN_1	int	1	網掛けパターン:001
		CLS_SHADED_PTN_2	int	2	網掛けパターン:002
		CLS_SHADED_PTN_3	int	3	網掛けパターン:003
		CLS_SHADED_PTN_4	int	4	網掛けパターン:004
		CLS_SHADED_PTN_5	int	5	網掛けパターン:005
		CLS_SHADED_PTN_6	int	6	網掛けパターン:006
		CLS_SHADED_PTN_7	int	7	網掛けパターン:007
		CLS_SHADED_PTN_8	int	8	網掛けパターン:008
		CLS_SHADED_PTN_9	int	9	網掛けパターン:009
		CLS_SHADED_PTN_10	int	10	網掛けパターン:010
		CLS_SHADED_PTN_11	int	11	網掛けパターン:011

21	単位	CLS_UNIT_MILLI	int	0	ミリ
		CLS_UNIT_INCH	int	1	インチ
22	速度設定	CLS_SPEEDSETTING_1	int	1	1:1.0 インチ(25.4mm)／秒
		CLS_SPEEDSETTING_2	int	2	2:2.0 インチ(50.8mm)／秒
		CLS_SPEEDSETTING_3	int	3	3:3.0 インチ(76.2mm)／秒
		CLS_SPEEDSETTING_4	int	4	4:4.0 インチ(101.6mm)／秒
		CLS_SPEEDSETTING_5	int	5	5:5.0 インチ(127.0mm)／秒
		CLS_SPEEDSETTING_6	int	6	6:6.0 インチ(152.4mm)／秒
		CLS_SPEEDSETTING_7	int	7	7:7.0 インチ(177.8mm)／秒
		CLS_SPEEDSETTING_8	int	8	8:8.0 インチ(203.2mm)／秒
		CLS_SPEEDSETTING_9	int	9	9:9.0 インチ(228.6mm)／秒
		CLS_SPEEDSETTING_A	int	10	A:1.0 インチ(25.4mm)／秒
		CLS_SPEEDSETTING_B	int	11	B:1.0 インチ(25.4mm)／秒
		CLS_SPEEDSETTING_C	int	12	C:2.0 インチ(50.8mm)／秒
		CLS_SPEEDSETTING_D	int	13	D:2.0 インチ(50.8mm)／秒
		CLS_SPEEDSETTING_E	int	14	E:3.0 インチ(76.2mm)／秒
		CLS_SPEEDSETTING_F	int	15	F:3.0 インチ(76.2mm)／秒
		CLS_SPEEDSETTING_G	int	16	G:4.0 インチ(101.6mm)／秒
		CLS_SPEEDSETTING_H	int	17	H:4.0 インチ(101.6mm)／秒
		CLS_SPEEDSETTING_I	int	18	I:5.0 インチ(127.0mm)／秒
		CLS_SPEEDSETTING_J	int	19	J:5.0 インチ(127.0mm)／秒
		CLS_SPEEDSETTING_K	int	20	K:6.0 インチ(152.4mm)／秒
		CLS_SPEEDSETTING_L	int	21	L:6.0 インチ(152.4mm)／秒
		CLS_SPEEDSETTING_M	int	22	M:7.0 インチ(177.8mm)／秒
		CLS_SPEEDSETTING_N	int	23	N:7.0 インチ(177.8mm)／秒
		CLS_SPEEDSETTING_O	int	24	O:8.0 インチ(203.2mm)／秒
		CLS_SPEEDSETTING_P	int	25	P:8.0 インチ(203.2mm)／秒
		CLS_SPEEDSETTING_Q	int	26	Q:9.0 インチ(228.6mm)／秒
		CLS_SPEEDSETTING_R	int	27	R:9.0 インチ(228.6mm)／秒
		CLS_SPEEDSETTING_S	int	28	S:10.0 インチ(254.0mm)／秒
		CLS_SPEEDSETTING_T	int	29	T:10.0 インチ(254.0mm)／秒
		CLS_SPEEDSETTING_U	int	30	U:11.0 インチ(279.4mm)／秒
		CLS_SPEEDSETTING_V	int	31	V:11.0 インチ(279.4mm)／秒
		CLS_SPEEDSETTING_W	int	32	W:12.0 インチ(304.8mm)／秒
		CLS_SPEEDSETTING_X	int	33	X:12.0 インチ(304.8mm)／秒
23	印刷後動作設定	CLS_MEDIAHANDLING_NONE	int	0	なし
		CLS_MEDIAHANDLING_TEAROFF	int	1	ティアオフ
		CLS_MEDIAHANDLING_DISPENSES	int	2	ディスペンス
		CLS_MEDIAHANDLING_PAUSE	int	3	ポーズ
		CLS_MEDIAHANDLING_CUT	int	4	カット
		CLS_MEDIAHANDLING_CUTANDPAUSE	int	5	カット、ポーズ
		CLS_MEDIAHANDLING_PEELOFF	int	6	剥離
		CLS_MEDIAHANDLING_REWIND	int	7	リワインダー
24	センサー選択 (透過/反射/なし)	CLS_SELSENSOR_NONE	int	0	なし
		CLS_SELSENSOR_SEETHROUGH	int	1	透過
		CLS_SELSENSOR_REFLECT	int	2	反射
25	印字方法	CLS_PRTMETHOD_TT	int	0	熱転写

	(TT/DT)	CLS_PRTMETHOD_DT	int	1	感熱
26	センサー選択 (前方/後方)	CLS_SENS_LOCATION_FRONT	int	0	フロントセンサ
		CLS_SENS_LOCATION_ADJUSTABLE	int	1	アジャスタブルセンサ

CITIZEN Windows Label Print SDK プログラムマニュアル

2024/04/12 Ver.1.05 用

シチズン・システムズ株式会社

<https://csj.citizen.co.jp/>