

# CITIZEN

Windows Layout SDK via POS Print SDK

プログラミング マニュアル

Version 1.4.0 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
2017.09.29	1.4.0.0	・新規

## 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但、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、本ライブラリをご使用いただけません。

## 著作権・商標について

- このプログラミング マニュアルの著作権は、シチズン・システムズ株式会社にあります。
  - CITIZEN は、シチズン時計株式会社の登録商標です。
  - Windows 及び Windows Server は、米国 Microsoft Corporation の米国およびその他の国における登録商標です。
  - Pentium は、米国およびその他の国における、Intel Corporation またはその子会社の商標または登録商標です。
  - QR コード、iQR コードは、(株)デンソーウェーブの登録商標です。
  - Android は、米国およびその他の国における Google Inc.の商標または登録商標です。
  - JavaScript は、米国およびその他の国における Oracle の商標または登録商標です。
- 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。

## 目次

更新履歴 .....	2
ご注意 .....	3
著作権・商標について .....	3
目次 .....	4
1. はじめに .....	5
1.1. ドキュメント対象範囲 .....	5
1.2. システム概要 .....	5
1.3. 対応 PC .....	6
1.4. 対応プリンター .....	6
1.5. 定義方法 .....	7
1.6. ライブラリ ファイル構成 .....	8
1.7. 機能一覧 .....	9
2. ライブラリ インターフェース .....	10
2.1. コンストラクタ .....	10
2.2. Open メソッド .....	11
2.3. Close メソッド .....	12
2.4. BeginPrint メソッド .....	13
2.5. EndPrint メソッド .....	14
2.6. DoPrint メソッド .....	15
2.7. InitFrame メソッド .....	17
2.8. SetPartsData メソッド .....	18
2.9. AddFrame メソッド .....	19
2.10. PrintOption クラス .....	20
3. コード サンプル ( C# ) .....	22
4. 注意事項 .....	23

## 1. はじめに

本ドキュメントは、Windows Layout SDK via POS Print SDK のプログラミング 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 レイアウト ファイルを利用する 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. システム概要

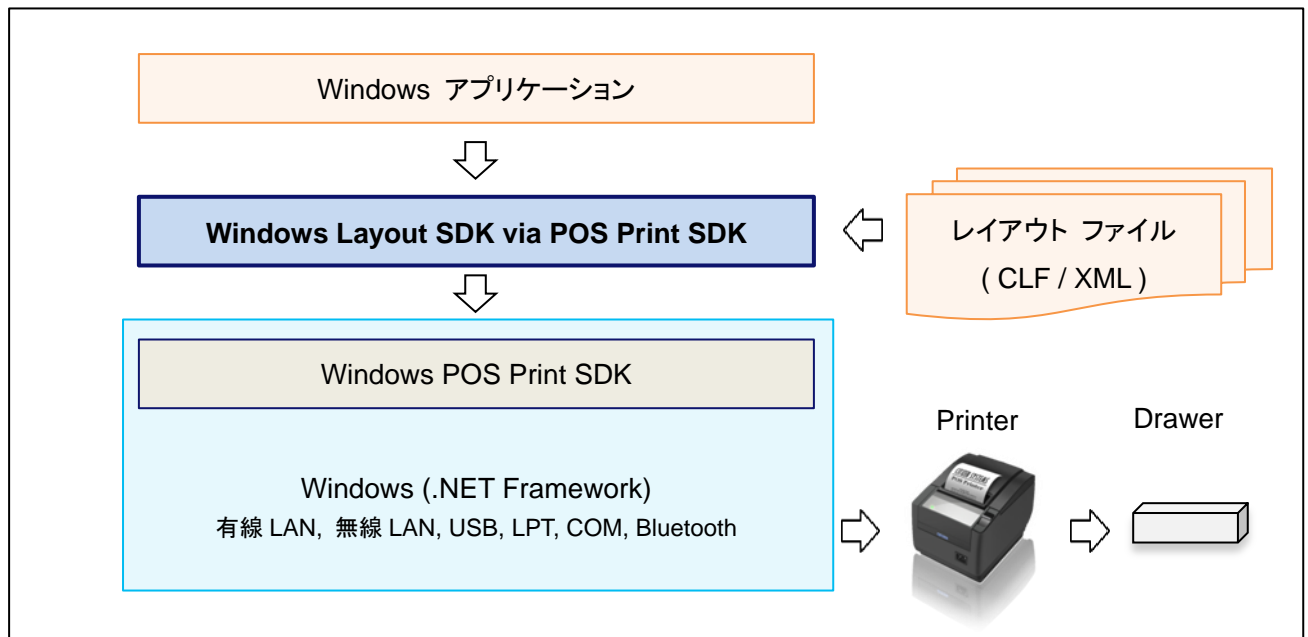
本ライブラリは、CITIZEN レイアウト ファイルを利用する Windows アプリケーションから参照されることを想定しています。

本ライブラリは、Windows POS Print SDK を介して印刷します。

本ライブラリは、CLF 形式と XML 形式の CITIZEN レイアウト ファイルに対応します。

CLF 形式は、PC 搭載フォントを利用して印刷する場合に利用します。

XML 形式は、プリンター搭載フォントを利用して印刷する場合に利用します。



ライブラリ システム 構成図

CITIZEN レイアウト ファイルは、Layout Editor で作成します。

Layout Editor の詳細については、「Layout SDK ユーザーズ・ガイド」をご参照してください。

### 1.3. 対応 PC

OS	<ul style="list-style-type: none"><li>• Windows XP</li><li>• Windows 7 (32bit, 64bit)</li><li>• Windows 8 (32bit, 64bit), Windows 8.1 (32bit, 64bit)</li><li>• Windows 10 (32bit, 64bit)</li></ul>
動作に必要なソフトウェア	<ul style="list-style-type: none"><li>• Microsoft .NET Framework 4.0</li><li>• Windows POS Print SDK Version 1.06.4 以降<sup>*1</sup></li></ul>

<sup>\*1</sup> : Windows POS Print SDK は、Layout SDK に含まれています。

「Windows プリンター ドライバー」を必要としません。

Windows プリンター ドライバーをインストールしている場合は、Windows プリンター ドライバーのポート設定の「**双方向サポートを有効にする**」のチェックを外す必要があります。

詳細については、「Windows POS Print SDK プログラミング マニュアル」をご参照ください。

### 1.4. 対応プリンター

本ライブラリの対象プリンターは、Windows POS Print SDK が対応するプリンターとなります。

各プリンターの機能詳細については、プリンターの取り扱い説明書をご参照ください。

## 1.5. 定義方法

### <ライブラリの追加>

1. Layout SDK ([LayoutSDK\_Setup\_ja]) をインストールします。  
詳細は、"Layout SDK ユーザーズ・ガイド" を参照してください。
2. インストール先から以下の 3 つのファイルを Visual Studio の [ソリューション エクスプローラー] の対象のプロジェクトに追加します。これらは、Windows POS Print SDK ライブラリ ファイルです。
  - CSJPOSLib.dll
  - CSJPOSLibW32.dll
  - CSJPOSLibW64.dll
3. 追加した 3 つのファイルの [プロパティ] ウィンドウの [出力ディレクトリにコピー] プロパティの [常にコピーする] を選択します。

### <参照の追加>

Visual C# で参照を追加するには、

1. [ソリューション エクスプローラー] で、プロジェクト ノードを右クリックし、[参照の追加] をクリックします。
2. [参照の追加] ダイアログ ボックスで、[参照] タブをクリックします。
3. 以下の 2 つのファイルを選択し、[OK] をクリックします。

[プロジェクトのフォルダー]¥CSJPOSLib.dll

C:¥Program Files¥CITIZEN¥Layout Utilities¥CSJSDKLayoutPrintEngine.dll

Visual Basic で参照を追加するには、

1. [ソリューション エクスプローラー] で、対象のプロジェクトの [My Project] ノードをダブルクリックします。
2. プロジェクト デザイナーで、[参照設定] タブをクリックします。
3. [追加] をクリックして [参照の追加] ダイアログ ボックスを開きます。
4. [参照の追加] ダイアログ ボックスで、[参照] タブをクリックします。
5. 以下の 2 つのファイルを選択し、[OK] をクリックします。

[プロジェクトのフォルダー]¥CSJPOSLib.dll

C:¥Program Files¥CITIZEN¥Layout Utilities¥CSJSDKLayoutPrintEngine.dll

### <名前空間の定義>

Visual C# の場合：

```
using com.citizen.sdk;
using com.citizen.sdk.poslayout;
```

Visual Basic の場合：

```
Imports com.citizen.sdk;
Imports com.citizen.sdk.poslayout;
```

## 1.6. ライブラリ ファイル構成

本ライブラリは、以下のファイルで構成されています。

- AxInterop.QRMAKERADLib.dll
- Citizen.LayoutUtilities.Common.dll
- **CSJPOSLib.dll**<sup>†</sup>
- CSJPOSLibW32.dll<sup>\*</sup>
- CSJPOSLibW64.dll<sup>\*</sup>
- **CSJSDKLayoutPrintEngine.dll**<sup>†</sup>
- GrapeSystems.Core.Common.dll
- GrapeSystems.Core.Drawing.Fx20.dll
- GrapeSystems.Core.Parts.dll
- GrapeSystems.Core.Parts.Frames.dll
- GrapeSystems.Library.BarcodeAd.dll
- GrapeSystems.Library.Controls.dll
- GrapeSystems.Library.Image.dll
- Interop.QRMAKERADLib.dll

<sup>\*</sup> :対象のプロジェクトにコピーが必要なファイルです。

<sup>†</sup> :対象のプロジェクトに[参照の追加]が必要なファイルです。



## 1.7. 機能一覧

本ライブラリは以下の機能を提供します。

### メソッド一覧

No.	名称	機能
1	LayoutPrintEngine	コンストラクタです。インスタンスを生成します。
2	Open	指定されたパス名の XML レイアウト ファイルを開きます。
3	Close	現在開いている XML レイアウト ファイルを閉じます。
4	BeginPrint	印刷の準備を開始します。
5	EndPrint	印刷データを破棄します。
6	DoPrint	指定されたプリンターで印刷を実行します。
7	InitFrame	印刷用データを設定するフレームを検索します。 指定されたフレーム名で検索し、内部で管理しているフレーム番号を返します。
8	SetPartsData	部品に印刷用データ(テキスト、バーコード、イメージ)を設定します。 InitFrame() で取得したフレーム番号および部品名で、設定対象とする部品を検索し、指定された文字列を設定します。
9	AddFrame	印刷用データが設定されたフレームを印刷対象として登録します。

### クラス

No.	名称	機能
10	PrintOption	印刷オプションです。DoPrint() の引数として指定します。

## 2. ライブラリ インターフェース

### 2.1. コンストラクタ

#### <定義>

```
com.citizen.sdk.poslayout.LayoutPrintEngine ()
```

#### <機能>

コンストラクタです。インスタンスを生成します。

#### <引数>

なし

#### <戻り値>

なし

#### <使用例>

```
com.citizen.sdk.poslayout.LayoutPrintEngine lpe =  
    new com.citizen.sdk.poslayout.LayoutPrintEngine();
```

## 2.2. Open メソッド

### <定義>

```
int Open ( string pathName )
```

### <機能>

指定されたパス名の CLF / XML レイアウト ファイルを開きます。

### <引数>

#### **pathName**

テンプレートとなる CLF / XML レイアウト ファイルのフルパス名を指定します。

### <戻り値>

- 0 : 正常に CLF / XML レイアウト ファイルを開くことができたことを意味します。
- 0 以外 : CLF / XML レイアウト ファイルが見つからないなど、何らかのエラーが発生したことを意味します。

### <使用例>

```
lpe.Open( "my_layout_File.CLF" );
```

## 2.3. Close メソッド

### <定義>

```
void Close ()
```

### <機能>

現在開いている CLF / XML レイアウト ファイルを閉じます。

### <引数・戻り値>

なし

### <使用例>

```
lpe.Close();
```

## 2.4. BeginPrint メソッド

### <定義>

```
void BeginPrint ()
```

### <機能>

印刷の準備を開始します。

### <引数・戻り値>

なし

### <使用例>

```
lpe.BeginPrint();
```

## 2.5. EndPrint メソッド

### <定義>

```
void EndPrint ()
```

### <機能>

印刷データを破棄します。

### <引数・戻り値>

なし

### <使用例>

```
lpe.EndPrint();
```

## 2.6. DoPrint メソッド

### <定義>

```
int DoPrint ( ESCPOSPrinter printer, boolean isReverse )
int DoPrint ( ESCPOSPrinter printer, boolean isReverse, PrintOption option )
int DoPrint ( ESCPOSPrinter printer, boolean isReverse, boolean isAlternativeFont )
int DoPrint ( ESCPOSPrinter printer, boolean isReverse, boolean isAlternativeFont, PrintOption
option )
```

### <機能>

指定されたプリンターで印刷を実行します。

—POINT—

ESCPOSPrinter.connect () メソッドにて接続が確立した ESCPOSPrinter オブジェクトを指定する必要があります。

### <引数>

#### **printer**

ESCPOSPrinter オブジェクトを指定します。

ESCPOSPrinter クラスは Windows POS Print SDK が提供するクラスです。

#### **isReverse**

上下反転印刷を指定します。

true : 上下反転(180 度回転)で逆順印刷(最終フレームから出力)します。

false : 正順印刷(先頭フレームから出力)します。

#### **isAlternativeFont**

代替フォントの使用を指定します。

**isAlternativeFont** が省略された場合は、false で印刷します。

CLF レイアウト ファイルのテキスト部品のみ有効です。

true : テキスト部品の [プリンターフォント] プロパティの設定で印刷します。

false : テキスト部品の [フォント] プロパティの設定で印刷します。

#### **option**

印刷オプションを指定します。詳細は、PrintOption クラスを参照してください。

**option** が省略された場合は、レイアウト ファイルの形式によって動作が異なります。

CLF 形式: PrintOption クラスの初期値で印刷します。

XML 形式: エクスポート時の [ドライバー設定] で印刷します。

## &lt;戻り値&gt;

- 0 : 正常終了したことを意味します。
- 0 超 : ESCPOSPrinter クラスのエラーコードです。印刷中に問題が発生したことを意味します。
- 0 未満 : 何らかのエラーが発生したことを意味します。

## &lt;使用例&gt;

```
com.citizen.sdk.ESCPOSPrinter printer = new com.citizen.sdk.ESCPOSPrinter ();
printer.setEncoding ( "ISO-8859-1" );
result = printer.Connect ( ESCPOSConst.CMP_PORT_WiFi, "192.168.10.100" );
if ( ESCPOSConst.CMP_SUCCESS == result ) {
    lpe.DoPrint ( printer, false, false, option );
    printer.disconnect ();
}
```



## 2.7. InitFrame メソッド

### <定義>

```
int InitFrame ( string frameName )
```

### <機能>

印刷用データを設定するフレームを検索します。

指定されたフレーム名で検索し、内部で管理しているフレーム番号を返します。

### <引数>

#### **frameName**

対象とするフレーム名を指定します。

### <戻り値>

0 以上 : 内部で管理しているフレーム番号を意味します。

-1 : 指定されたフレームが見つからないなど、何らかのエラーが発生したことを意味します。

### <使用例>

```
int frameIndex = lpe.InitFrame( "Frame1" );
```

## 2.8. SetPartsData メソッド

### <定義>

```
int SetPartsData ( int frameIndex, string partsName, string setText )
```

### <機能>

部品に印刷用データ(テキスト、バーコード、イメージ)を設定します。  
InitFrame()で取得したフレーム番号と部品名で、  
設定対象とする部品を検索し、指定された文字列を設定します。

### <引数>

#### **frameIndex**

対象とするフレーム番号 (InitFrame()で取得したもの)を指定します。

#### **partsName**

対象とする部品名を指定します。

#### **setText**

設定する印刷用データ(テキスト、バーコード、イメージ)を指定します。

### <戻り値>

- 0 : 正常終了したことを意味します。
- 0 以外 : 設定に失敗した場合など、何らかのエラーが発生したことを意味します。

### <使用例>

```
lpe.SetPartsData( frameIndex, "Text1", "New Text" );
```

```
lpe.SetPartsData( frameIndex, "Image1", "New Image File Path" );
```

## 2.9. AddFrame メソッド

### <定義>

```
int AddFrame ( int frameIndex )
```

### <機能>

印刷用データが設定されたフレームを、印刷対象として登録します。

### <引数>

#### **frameIndex**

対象とするフレーム番号 (`initFrame()` で取得したもの) を指定します。

### <戻り値>

- 0 以上 : 内部で管理している印刷登録時のフレーム番号を意味します。
- 1 : 登録に失敗した場合など、何らかのエラーが発生したことを意味します。

### <使用例>

```
lpe.AddFrame( frameIndex );
```

## 2.10. PrintOption クラス

### <定義>

`com.citizen.sdk.poslayout.PrintOption ()`

### <機能>

印刷オプションです。DoPrint() の引数となるクラスです。

### <引数・戻り値>

なし

### <クラス メンバー>

No.	名称	型	意味	設定可能範囲
1	MapMode	int	マッピング モード	CPE_MM_METRIC <sup>*</sup> : 0.01 ミリ単位 CPE_MM_ENGLISH : 0.001 インチ単位
2	Halftone	int	ハーフトーン	CPE_HT_THRESHOLD : Threshold CPE_HT_DITHER <sup>*</sup> : Dither
3	ColorMode	int	カラー印刷モード	CPE_CMD_MONO <sup>*</sup> : Monochrome CPE_CMD_GRAY <sup>*1</sup> : Grayscale
4	PaperMedia	int	用紙種類	CPE_PAPER_NORMAL <sup>*</sup> : Normal CPE_PAPER_LABEL_BM <sup>*2</sup> : Label/BM
5	CutterMode	int	用紙カット動作	CPE_CUT_NONE : No Cut CPE_CUT_PARTIAL <sup>*</sup> : Partial Cut CPE_CUT_FULL : Full Cut
6	PaperFeed	int	用紙送り	MapMode で定義された単位(デフォルトは 0.01 ミリ単位)で指定します。 <sup>*3</sup> 1270 <sup>*</sup>
7	DrawerOpen1	int	ドロワー 1 のタイミング	CPE_DRAWER_NEVER <sup>*</sup> : Never CPE_DRAWER_START : Print Start CPE_DRAWER_END : Print End
8	DrawerOpen2	int	ドロワー 2 のタイミング	CPE_DRAWER_NEVER <sup>*</sup> : Never CPE_DRAWER_START : Print Start CPE_DRAWER_END : Print End
9	PulseWidth1	int	ドロワー 1 のパルス信号幅	1 <sup>*</sup> ~ 8 ( x 100 ms ) <sup>*4</sup>
10	PulseWidth2	int	ドロワー 2 のパルス信号幅	1 <sup>*</sup> ~ 8 ( x 100 ms ) <sup>*5</sup>
11	BuzzerStart	int	印刷開始時のブザー回数	0 <sup>*</sup> ~ 9
12	BuzzerEnd	int	印刷終了時のブザー回数	0 <sup>*</sup> ~ 9

13	LogoStart	int	印刷開始時のロゴ印刷設定	CPE_LOGO_NONE <sup>*</sup> : None CPE_LOGO_PRINT01~20 <sup>*6</sup> : 1 ~ 20
14	LogoEnd	int	印刷終了時のロゴ印刷設定	CPE_LOGO_NONE <sup>*</sup> : None CPE_LOGO_PRINT01~20 <sup>*6</sup> : 1 ~ 20

<sup>\*</sup> : 初期値

<sup>\*1</sup> : 階調印刷対応プリンターが必要です。

<sup>\*2</sup> : ラベルまたはブラックマーク対応プリンターが必要です。

<sup>\*3</sup> : **CutterMode** に CPE\_CUT\_NONE を指定する必要があります。

<sup>\*4</sup> : **DrawerOpen1** に CPE\_DRAWER\_NEVER を指定した場合、**PulseWidth1** は無効です。

<sup>\*5</sup> : **DrawerOpen2** に CPE\_DRAWER\_NEVER を指定した場合、**PulseWidth2** は無効です。

<sup>\*6</sup> : あらかじめプリンターにロゴを登録する必要があります。

#### <使用例>

```
com.citizen.sdk.poslayout.PrintOption option = new com.citizen.sdk.poslayout.PrintOption();
option.CutterMode = LPEConst.CPE_CUT_PARTIAL;
option.DrawerOpen1 = LPEConst.CPE_DRAWER_START;
option.PulseWidth1 = 1;
option.LogoStart = LPEConst.CPE_LOGO_PRINT01;
```

### 3. コード サンプル ( C# )

```

com.citizen.sdk.poslayout.LayoutPrintEngine lpe =
    new com.citizen.sdk.poslayout.LayoutPrintEngine();           // Windows Layout SDK

com.citizen.sdk.poslayout.PrintOption option = new com.citizen.sdk.poslayout.PrintOption ();
option.CutterMode = LPEConst.CPE_CUT_PARTIAL;
option.DrawerOpen1 = LPEConst.CPE_DRAWER_START;
option.PulseWidth1 = 1;
option.LogoStart = LPEConst.CPE_LOGO_PRINT01;

int result = lpe.Open( "CLF / XML Layout File Name" );
if ( LPEConst.CPE_SUCCESS == result ) {
    lpe.BeginPrint();

    int frameIndex = lpe.InitFrame( "Frame1" );
    lpe.SetPartsData( frameIndex, "Text1", "New Text" );
    lpe.AddFrame( frameIndex );

    com.citizen.sdk.ESCPOSPrinter printer = new com.citizen.sdk.ESCPOSPrinter();
    printer.SetEncoding( "Shift_JIS" );
    result = printer.Connect( ESCPOSConst.CMP_PORT_WiFi, "192.168.182.100" );
    if ( ESCPOSConst.CMP_SUCCESS == result ) {
        lpe.DoPrint( printer, false, false, option );
        printer.Disconnect();
    }

    lpe.EndPrint();
    lpe.Close();
}

```

#### －POINT－

詳しくはレイアウト SDK サンプル プログラムをご参照ください。

<http://www.citizen-systems.co.jp/support/download/printer/sdk/index.html>

## 4. 注意事項

本ライブラリの注意事項を以下に示します。

### 3.1. 印刷完了確認について

本ライブラリは、Windows POS Print SDK が提供する印刷完了確認機能を利用しています。

印刷完了確認機能の詳細については、「Windows POS Print SDK プログラミング マニュアル」をご参照ください。

Windows Layout SDK via POS Print SDK

プログラミング マニュアル

Version 1.4.0 用