

CITIZEN

JavaScript Layout SDK

プログラミング マニュアル

Version 1.3.0 用

シチズン・システムズ株式会社

更新履歴

年月日	履歴
2016.08.16	1.3.0.0 リリース(新規)
2023.06.21	ドキュメント更新 - 印刷幅設定について を追記 - コード サンプル (JavaScript) に SetRecLineWidth() を追記

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 JavaScript 端末	6
1.4. 対応プリンター.....	6
1.5. 定義方法.....	6
1.6. 機能一覧.....	7
2. ライブラリ インターフェース	8
2.1. コンストラクタ.....	8
2.2. openFilePath.....	9
2.3. openDomObj.....	10
2.4. close.....	11
2.5. beginPrint.....	12
2.6. endPrint.....	13
2.7. preSend.....	14
2.8. initFrame	15
2.9. setPartsData	16
2.10. setImageData	17
2.11. addFrame	18
3. コード サンプル (JavaScript).....	19
4. 注意事項	21
4.1. 印刷完了確認について.....	21
4.2. 印刷幅設定について	21

1. はじめに

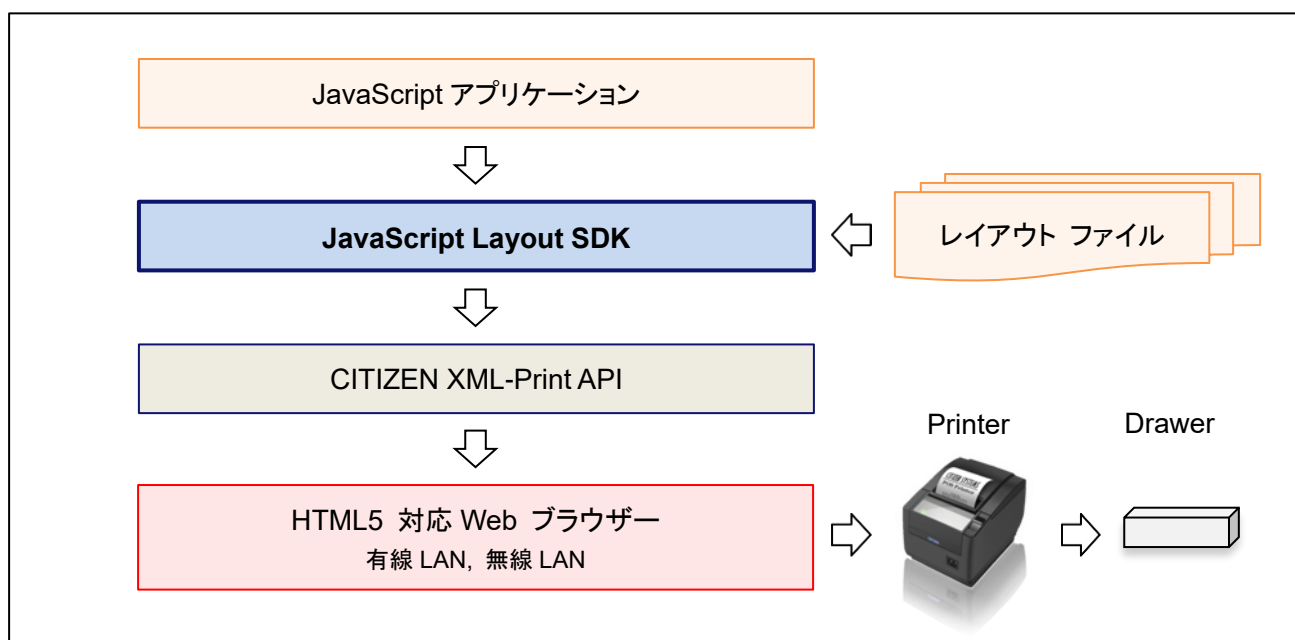
本ドキュメントは、JavaScript Layout SDK のプログラミング マニュアルです。

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 Android 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 JavaScript 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。



ライブラリ システム 構成図

1.3. 対応 JavaScript 端末

Web ブラウザー	HTML5 対応
動作に必要なソフトウェア	CITIZEN XML-Print API Version 1.02 以降 jQuery Version 1.1.4 以降

CITIZEN XML Print API の詳細については、「CITIZEN XML Print サービス プログラミング マニュアル」をご参照ください。

jQuery は、JavaScript Layout SDK が利用します。

1.4. 対応プリンター

本ライブラリの対象プリンターは、CITIZEN XML-Print サービス 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

本ライブラリを使用する場合は、提供ファイルをライブラリ追加し、以下のクラスをインポートしてください。

```
<script type="text/javascript" src="PrintEngine.js"></script>           // JavaScript Layout SDK
<script type="text/javascript" src="jquery-1.7.1.min.js"></script>      // jQuery
<script type="text/javascript" src="cxmip-api.js"></script>             // CITIZEN XML-Print API
```

1.6. 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.	メソッド	機能
1	PrintEngine	コンストラクタです。インスタンスを生成します。
2	openFilePath	レイアウトファイルを Web サーバーから読み込み、コールバック関数を呼び出します。
2	openDomObj	レイアウトファイルからパースした DOM オブジェクト(Document 型)を読み込み、コールバック関数を呼び出します。
3	close	openFilePath() / openDomObj() メソッドで読み込んだレイアウトファイルの情報を破棄します。
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	preSend	印刷バッファを元にプリンターへの送信準備を行います。
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)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。
8	setPartsData	部品に印刷用データ(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。

プロパティ一覧

No.	プロパティ	機能
1	OnLoad	XML レイアウトファイルの読み込み完了時のコールバック関数を設定します。 コード サンプル (JavaScript) をご参照ください。

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>

```
PrintEngine ()
```

<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>

```
var engine = new citizen.PrintEngine ();
```


2.2. openFilePath

<定義>

```
int openFilePath ( filePath )
```

<機能>

レイアウトファイルを Web サーバーから読み込み、コールバック関数を呼び出します。

コールバック関数は OnLoad プロパティに事前に設定してください。コールバック関数の引数の情報によって読み込みの成否を確認することができます。

読み込んだレイアウトファイルの情報は、close() メソッドが呼び出されるまで PrintEngine クラス内で保持されます。

<引数>

filePath (String)

Web サーバー上の XML レイアウトファイルのフルパスを指定します。

<戻り値>

0 : 正常に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。^{*1}

0 未満 : レイアウトファイルの不正なフォーマットなど、何らかのエラーが発生したことを意味します。

^{*1} : 戻り値はイメージ部品の読み込んだ結果を反映していない場合があります。コールバック関数の引数で判定してください。

<使用例>

```
engine.openFilePath ( "my_layout_File.XML" );
```

2.3. openDomObj

<定義>

```
int openDomObj ( domObj )
```

<機能>

レイアウトファイルからパースした DOM オブジェクト (Document 型) を読み込み、コールバック関数を呼び出します。

コールバック関数は OnLoad プロパティに事前に設定してください。コールバック関数の引数の情報によって読み込みの成否を確認することができます。

読み込んだレイアウトファイルの情報は、close() メソッドが呼び出されるまで PrintEngine クラス内で保持されます。

<引数>

domObj (Document)

レイアウトファイルからパースした DOM オブジェクトを指定します。

<戻り値>

- 0 : 正常に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。^{*1}
- 0 未満 : レイアウトファイルの不正なフォーマットなど、何らかのエラーが発生したことを意味します。

^{*1} : 戻り値はイメージ部品の読み込んだ結果を反映していない場合があります。コールバック関数の引数で判定してください。

<使用例>

2.4. close

<定義>

```
void close ()
```

<機能>

`openFilePath()` / `openDomObj()` メソッドで読み込んだレイアウトファイルの情報を破棄します。

本メソッド呼出し後、`initFrame()` / `setPartsData()` / `addFrame()` / `preSend()` メソッドはエラーとなります。再度、`openFilePath()` または、`openDomObj()` メソッドを呼び出す必要があります。

<引数・戻り値>

なし

<使用例>

```
engine.close ();
```

2.5. beginPrint

<定義>

```
int beginPrint ()
```

<機能>

印刷レイアウトの作成準備を行います。

本メソッド呼出し後に行う、`initFrame()` / `setPartsData()` / `addFrame()` メソッドにて作成した印刷レイアウトは、`endPrint()` メソッドが呼び出されるまで `PrintEngine` クラス内で保持されます。

<引数・戻り値>

- 0 : 正常終了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。

<使用例>

```
engine.beginPrint ();
```

2.6. endPrint

<定義>

```
void endPrint ()
```

<機能>

印刷レイアウトを破棄します。

本メソッド呼出し後は、preSend() メソッドによる送信準備はできません。

再度、beginPrint() メソッドを呼び出し、印刷レイアウトを作成する必要があります。

<引数・戻り値>

なし

<使用例>

```
engine.endPrint ();
```

2.7. preSend

<定義>

```
int preSend ( cxp, isReverse )
```

<機能>

印刷バッファを元にプリンターへの送信準備を行います。

<引数>

cxp (CXMLPrint オブジェクト)

出力先プリンターに合わせ、必要な初期設定を施した CXMLPrint オブジェクトを指定します。

isReverse (Boolean)

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。

<戻り値>

0 : 正常終了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。

<使用例>

```
var cxp = new citizen.CXMLPrint ();  
engine.preSend ( cxp, false );  
cxp.Send ( "http://192.168.129.82:8080/" ); // Send print data
```

2.8. initFrame

<定義>

```
int initFrame ( 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 (String)

対象とするフレーム名を指定します。

<戻り値>

1 以上 : 内部で管理しているフレーム番号を意味します。

0 未満 : 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>

```
var frameIndex = engine.initFrame ( "Frame1" );
```

2.9. setPartsData

<定義>

```
int setPartsData ( frameIndex, partsName, setText )
```

<機能>

部品に印刷用データ(テキスト、バーコード)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、指定された文字列またはデータを設定します。

<引数>

frameIndex (int)

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 (String)

対象とする部品名を指定します。

setText (String)

設定する印刷用データ(文字列)を指定します。テキスト／バーコード部品に適用します。

<戻り値>

- 0 : 正常終了したことを意味します。
- 0 未満 : 指定された部品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>

```
engine.setPartsData ( frameIndex , "Text1" , "NewText" );
```


2.10. setImageData

<定義>

```
int setImageData ( frameIndex, partsName, imageObj )
```

<機能>

部品に印刷用データ(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、指定されたデータを設定します。

<引数>

frameIndex (int)

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 (String)

対象とする部品名を指定します。

imageObj (Image オブジェクト)

設定する印刷用データ (Image オブジェクト) を指定します。イメージ部品に適用します。

<戻り値>

- 0 : 正常終了したことを意味します。
- 0 未満 : 指定された部品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>

```
engine.setImageData ( frameIndex , "Image1", imageObj );
```

2.11. addFrame

<定義>

```
int addFrame ( 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 (int)

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 : 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。
- 0 未満 : 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>

```
engine.addFrame ( frameIndex );
```

3. コード サンプル (JavaScript)

```

<script type="text/javascript" src="jquery-1.7.1.min.js"></script>
<script type="text/javascript" src="cxmllp-api.js"></script>
<script type="text/javascript" src="PrintEngine.js"></script>

<script language="javascript" type="text/javascript">
    function doPrint() {
        var clpe = new citizen.PrintEngine();// JavaScript Layout SDK
        clpe.OnLoad = doPrintProcess;
        clpe.openFilePath( "my_layout_File.XML" );
    }
    function doPrintProcess(clpe, result ) {
        if ( 0 > result ) { return; }                // Layout file open error

        var cxp = new citizen.CXMLPrint();
        cxp.MessageID( "12345678" );
        cxp.OnError = function(res) { window.alert( res.ResponseCode ); };
        cxp.OnReceive = function(res) { window.alert( res.status ); };
        cxp.SetEncoding( "USA" );                    // Codepage PC437
        // cxp.SetEncoding( "Multilingual" );          // Codepage PC850
        // cxp.SetRecLineWidth( 384 );                  // MSW8-1: 384 dots (58 mm paper)
        // cxp.SetRecLineWidth( 576 );                  // MSW8-1: 576 dots (80 mm paper)
        // cxp.SetRecLineWidth( 832 );                  // MSW8-1: 832 dots (112 mm paper)

        clpe.beginPrint();
        var frameIndex = clpe.initFrame( "Frame1" );
        clpe.setPartsData( frameIndex , "Text1", "NewText" );
        clpe.addFrame( frameIndex );

        clpe.preSend( cxp, false );
        cxp.Send( "http://192.168.129.82:8080/" );    // Send print data

        clpe.endPrint();
        clpe.close();
    }
</script>

```

－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。

4.1. 印刷完了確認について

本ライブラリは、CITIZEN XML-Print API 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、「CITIZEN XML-Print サービス プログラミング マニュアル」をご参照ください。

4.2. 印刷幅設定について

印刷幅は、CITIZEN XML-Print API の `SetRecLineWidth()` メソッドで設定します。設定値には、プリンターの印字領域幅 MSW8-1 の値を指定します。MSW8-1 が 576 ドット(80mm 用紙)の場合は印刷幅の設定を省略できます。MSW8-1 については、各プリンターの取り扱い説明書をご参照ください。

MSW8-1 が 384 ドット(58mm 用紙)の場合の例

```
cxp.SetRecLineWidth( 384 );
```

MSW8-1 が 576 ドット(80mm 用紙)の場合の例(省略可能)

```
cxp.SetRecLineWidth( 576 );
```

MSW8-1 が 832 ドット(112mm 用紙)の場合の例

```
cxp.SetRecLineWidth( 832 );
```

JavaScript Layout SDK

プログラミング マニュアル

Version 1.3.0 用