

CITIZEN

CITIZEN XML Device Control サービス JavaScript Device Control SDK プログラムマニュアル Ver. 1.00 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
2018/05/31	1.00	初版
2018/10/25		<ul style="list-style-type: none"> ・対応プリンター機種に CT-S257 を追加 ・対応周辺機器に BC-NL3000U を追加
2019/03/07		<ul style="list-style-type: none"> ・対応プリンター機種に CT-S4500 を追加
2020/06/03		<ul style="list-style-type: none"> ・対応プリンター機種に CL-E300EX/303EX、CL-E321EX/331EX を追加
2020/07/08		<ul style="list-style-type: none"> ・「1.1.システム概要」の説明修正
2021/03/03		<ul style="list-style-type: none"> ・対応プリンター機種に CT-E601 を追加 ・HTTPS に関する説明を追加 ・XML タグ名称の変更 <ul style="list-style-type: none"> 旧) SetCursorPostistion, Postistion 新) SetCursorPosition, Position ・サービスバージョンの確認方法を追加
2023/11/21		<ul style="list-style-type: none"> ・対応プリンター機種に CT-S801III/851III を追加 (7 頁) ・対応周辺機器に DSP01-LT2/DSP02-LS2 を追加 (7 頁)
2023/12/28		<ul style="list-style-type: none"> ・対応プリンター機種に CL-S700III/703III を追加 (7 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、本ライブラリをご使用いただけません。

商標

Microsoft、Windows は米国およびその他の国における Microsoft Corporation の商標または登録商標です。
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。

本マニュアルは製品改良などの理由により、予告なく変更になる場合がありますのでご了承ください。
ご採用の際にはお手数ですが弊社まで最新の資料をご請求くださりますようお願いいたします。

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. システム概要	6
1.2. ドキュメント対象範囲	6
1.3. システム構成例	7
1.4. 対応プリンター機種	7
1.5. 対応周辺機器	7
1.5.1. ラインディスプレイ設定	8
1.5.2. バーコードスキャナー設定	8
2. XML Device Control メッセージ	9
2.1. 要求メッセージ	9
2.1.1. 送信方法・メッセージ構成	9
2.1.2. <i>DeviceRequest</i> タグ	9
2.2. 応答メッセージ	10
2.2.1. メッセージ構成	10
2.2.2. 要求結果の取得	10
2.2.3. エラーコード	11
2.3. デバイスのステータス取得	12
2.3.1. ラインディスプレイ情報 (Name="LineDisplay")	12
2.3.2. バーコードスキャナー情報 (Name="Scanner")	12
3. デバイス制御タグ	13
3.1. デバイス制御タグ一覧	13
3.1.1. メッセージ ID (MessageID タグ)	13
3.1.2. ラインディスプレイ制御 (LineDisplay タグ)	13
3.1.3. バーコードスキャナー制御 (Scanner タグ)	14
3.2. ラインディスプレイ制御タグ詳細	15
3.2.1. 文字表示 (DisplayText タグ)	15
3.2.2. 表示クリア (ClearDisplay タグ)	15
3.2.3. ディスプレイ点滅 (BlinkDisplay タグ)	16
3.2.4. スクリーンモード設定 (SetDisplayMode タグ)	16
3.2.5. ディスプレイ設定 (SetDisplayConfig タグ)	17
3.2.6. カーソル設定 (SetCursorPosition タグ)	17
3.2.7. カーソル移動 (MoveCursor タグ)	18
3.2.8. カーソル型の指定 (SetCursorType タグ)	18
3.2.9. デバイス初期化 (InitializeDisplay タグ)	19
3.2.10. 文字表示 (SetEncoding タグ)	20
3.2.11. 国際文字の指定 (SetInternationalCharacterSet タグ)	21
3.3. バーコードスキャナー制御タグ詳細	22
3.3.1. スキャンデータ取得 (GetScanData タグ)	22
3.3.2. スキャンデータ取得中止 (AbortScan タグ)	23
4. XML Device Control サービス設定	24
4.1. Web マネージャ	24
4.1.1. Service 設定画面	24
XML Device Control	24
XML Device Control / Line Display	25
XML Device Control / Scanner	25
4.1.2. Service Status 画面	25
5. JavaScript Device Control SDK	26
5.1. 動作環境	26
5.2. プログラミングガイド	26
5.2.1. SDK ファイルの配置	26
5.2.2. プログラム構成	26

5.2.3. オブジェクトの作成.....	27
5.2.4. 応答受信コールバック関数の設定.....	27
5.2.5. 送信エラーコールバック関数の設定.....	28
5.2.6. デバイス制御処理.....	28
5.2.7. 送信実行.....	29
5.3. デバイスのステータス取得.....	30
5.3.1. デバイス情報取得メソッド(<i>GetDeviceInfo</i>).....	30
5.3.2. 応答受信コールバック関数の設定.....	30
ラインディスプレイ情報 (Name="LineDisplay").....	31
バーコードスキャナー情報 (Name="Scanner").....	31
5.4. デバイス制御メソッド一覧.....	31
5.4.1. ラインディスプレイ制御(<i>CXMLDisplay</i> オブジェクト).....	31
5.4.2. バーコードスキャナー制御(<i>CXMLScanner</i> オブジェクト).....	31
5.5. ラインディスプレイ制御メソッド詳細.....	32
5.5.1. メッセージ ID (<i>MessageID</i>).....	32
5.5.2. 文字表示 (<i>DisplayText</i>).....	32
5.5.3. 表示クリア (<i>ClearDisplay</i>).....	33
5.5.4. ディスプレイ点滅 (<i>BlinkDisplay</i>).....	33
5.5.5. スクリーンモード設定 (<i>SetDisplayMode</i>).....	34
5.5.6. ディスプレイ設定 (<i>SetDisplayConfig</i>).....	35
5.5.7. カーソル設定 (<i>SetCursorPosition</i>).....	35
5.5.8. カーソル移動 (<i>MoveCursor</i>).....	36
5.5.9. カーソル型の指定 (<i>SetCursorType</i>).....	36
5.5.10. デバイス初期化 (<i>InitializeDisplay</i>).....	37
5.5.11. 文字エンコード指定 (<i>SetEncoding</i>).....	37
5.5.12. 国際文字の指定 (<i>SetInternationalCharacterSet</i>).....	38
5.6. バーコードスキャナー制御メソッド詳細.....	39
5.6.1. メッセージ ID (<i>MessageID</i>).....	39
5.6.2. 連続スキャン開始 (<i>StartScanning</i>).....	39
5.6.3. 連続スキャン停止 (<i>StopScanning</i>).....	40
5.6.4. スキャンデータ取得 (<i>GetScanData</i>).....	41
5.6.5. スキャンデータ取得中止 (<i>AbortScan</i>).....	41
5.7. SDK 設定/その他機能.....	42
5.7.1. SDK バージョン番号の取得 (<i>GetVersionCode</i>).....	42
5.7.2. SDK バージョン文字列の取得 (<i>GetVersionName</i>).....	42
6. サンプルプログラム.....	43
6.1. JavaScript Device Control SDK サンプル.....	43

1. はじめに

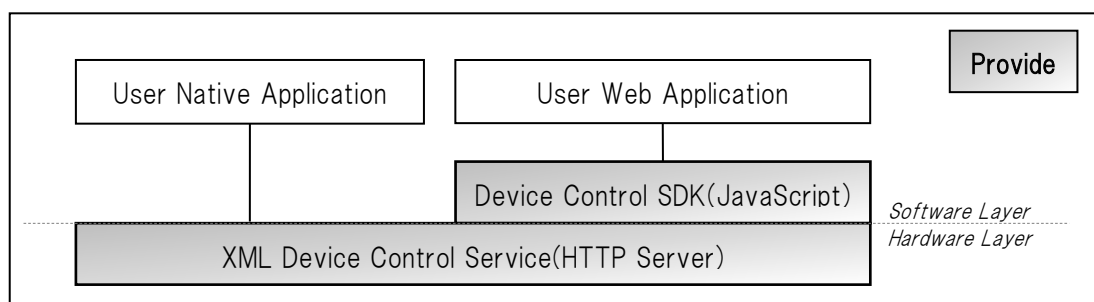
本ドキュメントは、CITIZEN XML Device Control サービスにおけるプログラマー様向けのマニュアルです。

1.1. システム概要

CITIZEN XML Device Control サービスは、OS 非依存のマルチプラットフォーム環境において、デバイスドライバーレスで、プリンターに接続した周辺機器を制御する機能を提供します。

プリンターの印刷制御をするには、CITIZEN XML Print サービス (JavaScript POS Print SDK / JavaScript Label Print SDK) をご利用ください。

HTTP(XML)ベースの制御方式のため、Web サービス環境から、周辺機器を簡単に制御できます。また、XML Device Control をクライアントサイドの JavaScript で制御するためのライブラリとして、Device Control SDK を用意しております。以下に、提供サービスの概念図を示します。



1.2. ドキュメント対象範囲

本ドキュメントは、CITIZEN XML Device Control サービス対応の周辺機器を利用するアプリケーション開発者が参照することを目的としています。

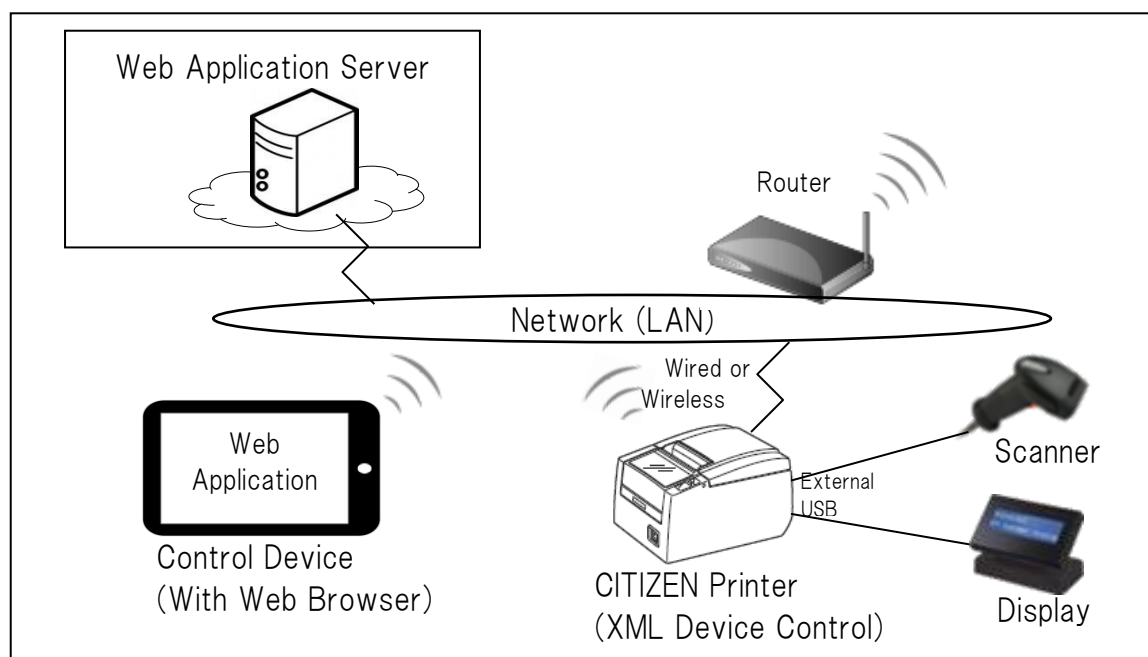
HTTP(XML)ベースで制御するための制御仕様は、本書の「[2. XML Device Control メッセージ](#)」、「[3. デバイス制御タグ](#)」、「[4. XML Device Control サービス設定](#)」を参照してください。

Web サービス環境から制御するための JavaScript SDK 仕様は、「[5. JavaScript Device Control SDK](#)」を参照してください。

本サービスの動作確認につきましては、「[6. サンプルプログラム](#)」を参照してください。

1.3. システム構成例

以下に、CITIZEN XML Device Control サービスにおけるシステム構成例を示します。



1.4. 対応プリンター機種

本サービスの対象プリンターの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、各取り扱い説明書をご参照ください。

対象プリンターモデル	インターフェース
CT-E601, CT-S251W/255W/257/4500, CL-E300EX/303EX/321EX/331EX	有線 LAN (型番: IF2-EFX2) 有線/無線 LAN (型番: IF2-WFx5, IF2-WFx6)
CT-S601IIW/651IIW/801IIW/851IIW,801III/851III, CL-E720/730, CL-S400DT/700III/703III	有線 LAN (型番: IF1-EFX2) 有線/無線 LAN (型番: IF1-WFx6)

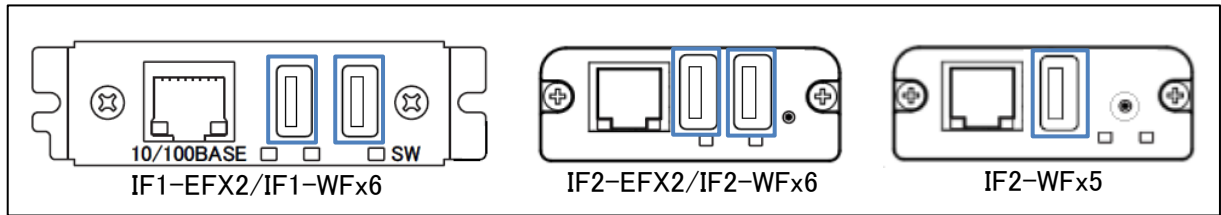
本サービスのバージョン番号は、「[4.1.2. Service Status 画面](#)」を確認してください。バージョンにより、利用方法や機能が異なる場合があります。

1.5. 対応周辺機器

本サービスの制御対象とする周辺機器のモデルは以下の通りです。各モデルの機能詳細については、各周辺機器の取り扱い説明書をご参照ください。

対象ディスプレイ	製品仕様概要
DSP01-LT/DSP01-LT2	TFT 型ラインディスプレイ
DSP02-LS/DSP02-LS2	STN 型ラインディスプレイ
対象スキャナー	製品仕様概要
SCN01-Z1D	1 次元バーコードスキャナー
SCN02-Z2D	2 次元バーコードスキャナー
BC-NL3000U	2 次元バーコードスキャナー

対象周辺機器の接続は、プリンター電源を一旦 OFF にしてから、下図に示すインターフェースの USB 端子に接続してご利用ください。その後、プリンター電源を ON にしてから、対象周辺機器が利用可能になるまで、安定動作のため、周辺機器の制御開始処理を 30 秒ほど待機してから実行してください。



以下は、周辺機器接続に関しての、してはいけない「禁止」内容になります。

禁止事項

- 対応周辺機器以外(USB ハブやスマートフォン等)をインターフェースの USB 端子に接続
- プリンター電源が ON のまま、インターフェースの USB 端子から周辺機器のケーブルを挿抜
- 同種の周辺機器をインターフェースの USB 端子に複数接続 (例:ディスプレイを 2 台接続)

もし、上記事項を実施された場合、プリンターや接続周辺機器について、誤動作を招く原因や、最悪、故障の原因となりますので、おやめください。

以下に、本サービスをご使用になる際の各周辺機器の設定情報を説明します。

1.5.1. ラインディスプレイ設定

通常は、ディスプレイの工場出荷状態のままで構いません。もし、標準の工場出荷設定と異なるディスプレイをご利用の場合は、「[4.1.1.Service 設定画面](#)」で、利用ディスプレイの仕様と合わせてください。

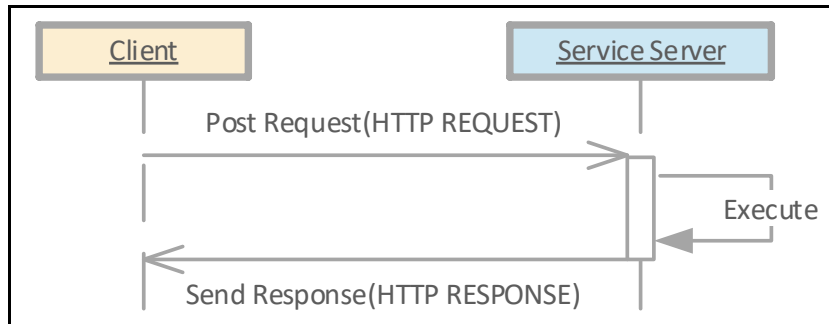
1.5.2. バーコードスキャナー設定

接続スキャナーが以下の通り設定されていることを確認してください。設定方法は、ご利用周辺機器の取り扱い説明書をご参照ください。

項目	値	説明
Interface	USB HID Class	通信プロトコル
Keyboard	US Keyboard	キーボード言語
Terminator	Enter	データのサフィックス

2. XML Device Control メッセージ

下図のように、サービス利用者は、要求メッセージを HTTP のリクエストとして発行し、サービスからの応答メッセージは、HTTP のレスポンスとして受信します。要求メッセージおよび応答メッセージは、XML スキーマファイル「CitizenXML-Device.xsd」で定義されています。



2.1. 要求メッセージ

クライアントから発行される要求メッセージに応じて、デバイス(周辺機器)の制御を行います。

2.1.1. 送信方法・メッセージ構成

要求メッセージは、次の方法で、SOAP メッセージを送信してください。

- ・ HTTP URL: http(s):// [本サービスの IP アドレス] /xmldevice
 ※サービスバージョン 1.0 以前では、http:// [本サービスの IP アドレス] :8085
- ・ HTTP メソッド: POST
- ・ HTTP ヘッダー: Content-Type: text/xml; charset=UTF-8

要求メッセージの構成は、以下の通りです。

```

<?xml version="1.0" encoding="UTF-8"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>

    <DeviceRequest xmlns="http://www.citizen.co.jp/DeviceControl/"
      MajorVersion="1">
      <!-- 要求 ID -->
      <MessageID>a55f0b87-4bb1-94fb-a92b-dfa2913c4343</MessageID>

      <!-- ラインディスプレイ制御-->
      <LineDisplay>
        <ClearDisplay/>
        <SetCursorPosition>
          <Position>1,1</Position>
        </SetCursorPosition>
        <DisplayText>
          <Data>Display\r\nText</Data>
        </DisplayText>
      </LineDisplay>

    </DeviceRequest>

  </s:Body>
</s:Envelope>
  
```

<DeviceRequest>タグ

2.1.2. DeviceRequest タグ

デバイスの制御は、要求メッセージ中の<DeviceRequest>タグ内にデバイス制御タグを記述してください。デバイス制御タグの詳細は、本書の「[3. デバイス制御タグ](#)」を参照してください。

2.2. 応答メッセージ

本サービスからの応答メッセージにより、要求の結果を確認できます。

2.2.1. メッセージ構成

応答メッセージの構成は、以下の通りです。

```
<?xml version="1.0" encoding="UTF-8"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>

    <DeviceResponse xmlns="http://www.citizen.co.jp/DeviceControl/"
      MajorVersion="1">
      <MessageID>7338ab39-92f2-416b-ac00-25b21956142e</MessageID>
      <Response ResponseCode="OK">
        <RequestID>a55f0b87-4bb1-94fb-a92b-dfa2913c4343</RequestID>
      </Response>
    </DeviceResponse>

  </s:Body>
</s:Envelope>
```

} <DeviceResponse>タグ

2.2.2. 要求結果の取得

<DeviceResponse>タグ内の情報でリクエストの結果を確認できます。

項目	説明
ResponseCode 属性	処理結果が格納されます
MessageID 要素	応答メッセージを識別のための ID が格納されます
RequestID 要素	送信時に指定された MessageID が格納されます
BusinessError 要素	エラー発生時にエラー情報が格納されます

エラーの発生の確認

Response 要素の ResponseCode 属性の値を確認することにより、エラーの発生の有無が確認できます。

コード	説明
OK	正常に終了した
Rejected	エラーが発生した

以下は、正常に終了した場合の Response 要素の例です。

```
<Response ResponseCode="OK">
  <RequestID>a55f0b87-4bb1-94fb-a92b-dfa2913c4343</RequestID>
</Response>
```

エラー情報の確認

エラー発生時には、Response 要素内の BusinessError 要素内の Code および Description 要素の内容により、結果の要因を確認できます。詳細は、本書の「[2.2.3 エラーコード](#)」を参照してください。

以下は、エラーが発生した場合の Response 要素の例です。

```
<Response ResponseCode="Rejected">
  <RequestID>a55f0b87-4bb1-94fb-a92b-dfa2913c4343</RequestID>
  <BusinessError Severity="Error">
    <Code>ENotConnect</Code>
    <Description>Failed connection to the device.</Description>
  </BusinessError>
</Response>
```

2.2.3. エラーコード

応答メッセージでは、エラーコードやエラーの説明等の詳細情報を、Response 要素内の BusinessException 要素に設定されます。本サービスで使用するエラーコードを以下に示します。

サービス処理中に発生するエラー

コード	説明
RequestInvalid	要求情報が不正
DeviceTimeout	デバイスはタイムアウトした

デバイス処理中に発生するエラー

コード	説明
ENotConnect	デバイス接続失敗
EConnectNotFound	対応機種でない
Ellegal	未サポートまたは無効パラメータ
EOffline	デバイスがオフライン
ENoExist	データ無しエラー
EFailure	処理が実行できない
ETimeout	処理タイムアウト
EptrBadFormat	データ書式エラー
EptrTooBig	データサイズエラー

Description は、Response 要素内の BusinessException 要素内に設定されます。以下は、一例です。

```
<BusinessError Severity="Error">
  <Code>ENotConnect</Code>
  <Description>Failed connection to the device.</Description>
</BusinessError>
```

2.3. デバイスのステータス取得

デバイスのステータスを取得するには、デバイス情報取得要求を指定する GetDeviceInfo タグを使用します。

パラメータ

属性	意味	設定可能範囲
Name	デバイス名	対象デバイス名 指定されなかった場合は全登録デバイスが対象
Status	ステータス情報フラグ	"true"を指定した場合にステータス情報を付加

要求メッセージの例は、以下の通りです。

```
<?xml version="1.0" encoding="UTF-8"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <DeviceRequest xmlns="http://www.citizen.co.jp/DeviceControl/"
      MajorVersion="1">
      <!-- デバイス情報取得要求 -->
      <GetDeviceInfo Status="true"/>
    </DeviceRequest>
  </s:Body>
</s:Envelope>
```

応答メッセージの例は、以下の通りです。

```
<?xml version="1.0" encoding="UTF-8"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <DeviceResponse xmlns="http://www.citizen.co.jp/DeviceControl/"
      MajorVersion="1">
      <MessageID>5e60c4c2-c0a4-44a2-83ee-2984a1c620a0</MessageID>
      <Response ResponseCode="OK">
        <RequestID />
      </Response>
      <GetDeviceInfo>
        <Device Name="LineDisplay" Status="Online" />
        <Device Name="Scanner" Status="Online" />
      </GetDeviceInfo>
    </DeviceResponse>
  </s:Body>
</s:Envelope>
```

<GetDeviceInfo>タグ

デバイス情報の内容は、応答メッセージの Device 要素の Status 属性にステータスコードが格納されます。
本サービスで使用するステータスコードを以下に示します。

2.3.1. ラインディスプレイ情報 (Name="LineDisplay")

ステータスコード	説明
Online	利用可能な状態です。
EConnectNotFound	本サービスでサポートされないデバイスが接続されています
ENotConnect	ラインディスプレイが接続されていません。
EOffline	他サービスでデバイスを利用中です。

2.3.2. バーコードスキャナー情報 (Name="Scanner")

ステータスコード	説明
Online	利用可能な状態です。
EConnectNotFound	本サービスでサポートされないデバイスが接続されています
ENotConnect	バーコードスキャナーが接続されていません。
EOffline	他サービスでデバイスを利用中です。
StatusScanning	デバイスからのデータ受信待ち状態です。

3. デバイス制御タグ

3.1. デバイス制御タグ一覧

本サービスで利用できるデバイス制御タグを以下に示します。

分類	機能	タグ	説明
全般	メッセージ ID	<MessageID>	発信者がメッセージを識別するために指定します。
デバイス制御	ラインディスプレイ制御	<Display>	ラインディスプレイを制御するために指定します。
	バーコードスキャナー制御	<Scanner>	バーコードスキャナーを制御するために指定します。

3.1.1. メッセージ ID (MessageID タグ)

値

要求メッセージ ID を指定します。

説明

このタグは、発信者がメッセージを識別するために使用します。

指定された要求メッセージ ID が応答メッセージの[<RequestID>](#)タグに付加されます。応答メッセージの詳細は本書の[「2.2. 応答メッセージ」](#)を参照してください。

使用例

```
<MessageID>12345678</MessageID>
```

3.1.2. ラインディスプレイ制御 (LineDisplay タグ)

値

LineDisplay タグ内に、以下の制御タグが記述できます。

機能	タグ	説明
文字列表示	<DisplayText>	文字を表示します。
表示クリア	<ClearDisplay>	表示文字を消去します。
点滅表示	<BlinkDisplay>	点滅表示させます。
ディスプレイ設定	<SetDisplayMode>	ディスプレイモードを設定します。
	<SetDisplayConfig>	各種設定を行います。
カーソル設定	<SetCursorPosition>	カーソルの位置指定をします。
	<MoveCursor>	カーソルを移動します。
	<SetCursorType>	カーソル位置を表示します。
初期化	<InitializeDisplay>	ディスプレイを初期化します。
文字列表示設定	<SetEncode>	送信する文字列のエンコードを指定します。
	<SetInternationalCharacterSet>	国際文字を指定します。

説明

このタグは、ラインディスプレイを制御するために使用します。詳細は本書の[「3.2. ラインディスプレイ制御タグ詳細」](#)を参照してください。

3.1.3. バーコードスキャナー制御 (Scanner タグ)

値

Scanner タグ内に、以下の制御タグが記述できます。

機能	タグ	説明
データ取得	< GetScanData >	データを取得します。
データ取得中止	< AbortScan >	データ取得を中止します。

説明

このタグは、バーコードスキャナーを制御するために使用します。詳細は本書の「[3.3.バーコードスキャナー制御タグ詳細](#)」を参照してください。

3.2. ラインディスプレイ制御タグ詳細

3.2.1. 文字表示(DisplayText タグ)

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。

要素	意味	設定可能範囲
Data	テキストデータ	
Attribute	テキスト属性	0: 標準 64: 反転 (省略可能要素, 省略時は 0)

説明

このタグは、現在のカーソル位置からテキストを表示するために使用します。
テキスト属性は、反転を指定可能です。

使用例

```
<LineDisplay>  
  <DisplayText>  
    <Data>Hello, World!</Data>  
    <Attribute>0</Attribute>  
  </DisplayText>  
</LineDisplay>
```

3.2.2. 表示クリア(ClearDisplay タグ)

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。

要素	意味	設定可能範囲
DisplayArea	消去領域	All: 全領域 CursorLine: カーソル行 (省略可能要素, 省略時は All)

説明

このタグは、表示中の文字を消去します。

使用例

```
<LineDisplay>  
  <ClearDisplay>  
    <DisplayArea>All</DisplayArea>  
  </ClearDisplay>  
</LineDisplay>
```

3.2.3. ディスプレイ点滅 (BlinkDisplay タグ)

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。

要素	意味	設定可能範囲
IntervalBlink	点滅間隔(ミリ秒)	0～

説明

このタグは、表示画面全体を点滅させます。

点滅間隔(ミリ秒)は、点灯と消灯の間隔を指定します。点滅間隔に、0 を指定すると、点滅は解除されます。

使用例

```
<LineDisplay>
  <BlinkDisplay>
    <IntervalBlink>1000</IntervalBlink>
  </BlinkDisplay>
</LineDisplay>
```

3.2.4. スクリーンモード設定 (SetDisplayMode タグ)

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。

要素	意味	設定可能範囲
DisplayMode	ディスプレイモード	Overwrite : 上書きモード VerticalScroll : 垂直スクロールモード HorizontalScroll : 水平スクロールモード

説明

以下のディスプレイのモードを設定します。

DisplayMode	概要
Overwrite	カーソル位置の文字を上書きし、カーソルを右に移動。 (カーソルが上端右端のときの入力は、カーソルを下端左端に移動、 カーソルが下端右端のときの文字入力は、カーソルを上端左端に移動)
VerticalScroll	カーソルが上端のときのカーソル上移動(または、左端での左移動)で、上端の表示行を下端にスクロール カーソルが下端のときのカーソル下移動(または、右端での右移動)で、下端の表示行を上端にスクロール
HorizontalScroll	カーソルが右端でのカーソル右移動(または、文字入力)で、現在のカーソル行に対して、左方向に文字をスクロール カーソル左端のカーソル左移動で、現在のカーソル行に対して、右方向に文字をスクロール

使用例

```
<LineDisplay>
  <SetDisplayMode>
    <DisplayMode>HorizontalScroll</DisplayMode>
  </SetDisplayMode>
</LineDisplay>
```


3.2.5. ディスプレイ設定 (SetDisplayConfig タグ)

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。

要素	意味	設定可能範囲
Brightness	輝度 (%)	0～100

説明

このタグは、表示画面の輝度を変更します。

輝度は、数値が大きいほど明るくなります。0 を指定すると、画面を消灯します(表示内容は保持されます)。

設定後、表示画面全体の点滅は解除されます。

使用例

```
<LineDisplay>  
  <SetDisplayConfig>  
    <Brightness>40</Brightness>  
  </SetDisplayConfig>  
</LineDisplay>
```

3.2.6. カーソル設定 (SetCursorPosition タグ)

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。

要素	意味	設定可能範囲
Position	カーソル位置	1～、1～

説明

このタグは、カーソルを設定するために使用します。

カーソル位置は、カーソルの移動座標で、カンマ区切りの 2 つの ASCII 数字のみで構成されます。構成は、桁位置、行位置、の順に列挙します。

使用例

```
<LineDisplay>  
  <SetCursorPosition>  
    <Position>1,2</Position>  
  </SetCursorPosition>  
</LineDisplay>
```

3.2.7. カーソル移動 (MoveCursor タグ)

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。

要素	意味	設定可能範囲
Movement	カーソル移動量	-128～127、-128～127

説明

このタグは、カーソルを移動するために使用します。

現在のカーソル位置からの移動となります。カーソル移動量は、カンマ区切りの 2 つの ASCII 数字のみで構成されます。構成は、左右方向移動量(-:左方向,+:右方向)、上下方向移動量(-:上方向,+:下方向)、の順に列挙します。

使用例

```
<LineDisplay>
  <MoveCursor>
    <Movement>0,1</Movement>
  </MoveCursor>

  <MoveCursor>
    <Movement>-1,0</Movement>
  </MoveCursor>
</LineDisplay>
```

3.2.8. カーソル型の指定 (SetCursorType タグ)

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。

要素	意味	設定可能範囲
CursorType	カーソル型指定	None: カーソル非表示 UnderLine: カーソル表示 (省略可能要素, 省略時は UnderLine)

説明

現在のカーソル位置をディスプレイに表示します。

使用例

```
<LineDisplay>
  <SetCursorType>
    <CursorType>UnderLine</CursorType>
  </SetCursorType>
</LineDisplay>
```

3.2.9. デバイス初期化 (InitializeDisplay タグ)

パラメータ

ありません。

説明

デバイスを初期化します。

使用例

```
<LineDisplay>  
  <InitializeDisplay/>  
</LineDisplay>
```

3.2.10. 文字表示(SetEncoding タグ)

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。

要素	意味	設定可能範囲
Encode	文字エンコード	SingleByteCharacter: 8-bit Character Japanese: 日本語(katakana) SimplifiedChinese: 簡体字中国語 Korean: 韓国語 TraditionalChinese: 繁体字中国語 None: 無変換

説明

DisplayText タグ内のテキストデータ(要素 Data/UTF-8 形式)をデバイスに送信するときの文字エンコードを設定します。この設定の初期値は、“SingleByteCharacter”です。

使用例

```

<LineDisplay>
  <SetEncoding>
    <Encode>Japanese</Encode>
  </SetEncoding>
  <DisplayText>
    <Data>コンニチワ</Data>
  </DisplayText>
</LineDisplay>

```

3.2.11. 国際文字の指定 (SetInternationalCharacterSet タグ)

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。

要素	意味	設定可能範囲
CharacterSet	国際文字指定	0～16

説明

以下の国際文字セットを設定します。

CharacterSet	国際文字セット	CharacterSet	国際文字セット
0	アメリカ	9	ノルウェー
1	フランス	10	デンマークⅡ
2	ドイツ	11	スペインⅡ
3	イギリス	12	ラテンアメリカ
4	デンマークⅠ	13	韓国
5	スウェーデン	14	クロアチア
6	イタリア	15	中国
7	スペインⅠ	16	ベトナム
8	日本		

使用例

```
<LineDisplay>
  <SetInternationalCharacterSet>
    <CharacterSet>8</CharacterSet>
  </SetInternationalCharacterSet>
  <SetEncoding>
    <Encode>Japanese</Encode>
  </SetEncoding>
  <DisplayText>
    <Data>Total:¥¥1,010</Data>
  </DisplayText>
</LineDisplay>
```

3.3. バーコードスキャナー制御タグ詳細

3.3.1. スキャンデータ取得(GetScanData タグ)

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。

要素	意味	設定可能範囲
Timeout	タイムアウト(秒)	0～60

応答メッセージ

応答メッセージの意味は以下の通りです。

要素	意味	補足
Data	スキャンデータ	

説明

このタグは、デバイスからデータを取得するために使用します。取得したデータは、応答メッセージの Data 要素で応答されます。タイムアウトに指定した時間、デバイスからデータを待ち受けます。もし、タイムアウトに指定した時間待っても、デバイスからデータを受信しなかった場合は、エラー(ETimeout)を応答します。

このタグの呼び出し前に、デバイスから受信したデータがあれば蓄積します。その場合は、本タグ呼び出し時に、蓄積されたデータ群を応答します。また、蓄積されたデータが無い場合は、新たなデータをデバイスから受信した段階で、タイムアウト時間を待たずに、受信データを即時に応答します。

使用例

```
<Scanner>
  <GetScanData>
    <Timeout>10</Timeout>
  </GetScanData>
</Scanner>
```

～応答メッセージ～

```
<DeviceResponse>
  ...
  <Scanner>
    <GetScanData>
      <Data>CTJV052985</Data>
    </GetScanData>
  </Scanner>
  ...
```

～応答メッセージ～ 2 つ以上のスキャンデータが有る場合 (最初の Data タグが一番古い)

```
<DeviceResponse>
  ...
  <Scanner>
    <GetScanData>
      <Data>CTJV052985</Data>
      <Data>0123456789</Data>
      ...
      <Data>ABCDEFGHIJ</Data>
    </GetScanData>
  </Scanner>
  ...
```

OLD
↓
NEW

3.3.2. スキャンデータ取得中止 (AbortScan タグ)

パラメータ

ありません。

説明

このタグは、デバイスからのスキャンデータ取得待ち処理を中止します。

使用例

```
<Scanner>  
  <AbortScan/>  
</Scanner>
```

4. XML Device Control サービス設定

CITIZEN XML Device Control サービスの設定方法を説明します。

4.1. Web マネージャ

Web ブラウザーから各プリンターに接続することでデバイスに関する設定を変更することができます。基本的な操作につきましては、プリンターのインターフェイスボード取り扱い説明書をご参照ください。

本ドキュメントでは、XML Device Control サービスの設定項目を説明します。

4.1.1. Service 設定画面

以下の Service 画面で、プリンターが提供するサービスの設定を行います。

この画面は、XML Device Control サービスに未対応のプリンターでは表示されませんので、対応プリンターをご利用ください。

各設定値は、電源を切断しても値を保持します。工場初期設定(Factory Default)の処理が行われた時には、各設定値を初期値に設定します。

XML Device Control

以下の XML Device Control サービスに関する一般設定を行います。

項目	初期値	設定範囲	説明
Port Number	8085	1025～65535	接続ポート番号 ※旧仕様互換のため設定は不要です
Timeout for connect	10	5～180	制御開始待ちのタイムアウト時間(秒)
Max Connections	2	1～3	最大同時接続数(通常は初期値で利用)

XML Device Control / Line Display

以下のディスプレイに関する一般設定を行います。設定初期値は、既に対応ディスプレイに対して適切値になっているので、通常利用では変更しないでください。

項目	初期値	設定範囲
Baud rate	9600	2400, 4800, 9600, 19200, 38400, 57600, 115200
Data	8 bit	7 bit, 8 bit
Parity	None	None, Odd, Even
Stop	1 bit	1 bit , 2 bit
Flow Control	Off	Hardware, Xon/Xoff, Off

設定変更後は、画面下部の”Submit”ボタンを押し、Maintenance メニューの”Save & Reboot”ボタンを押し、ボード再起動後に有効になります。

“Test Device”ボタンを押すと、本設定に従ってディスプレイにテスト文字列を表示します。ディスプレイと接続を確認できない場合は、アラートメッセージ(“Test failed”)をブラウザに表示します。

XML Device Control / Scanner

“Test Device”ボタンを押すと、スキャナー(USB HID Keyboard 方式)との接続を確認します。スキャナーとの接続を確認できない場合は、アラートメッセージ(“Test failed”)をブラウザに表示します。

4.1.2. Service Status 画面

Service Status 画面では、サービスのバージョン情報や設定、周辺機器状態を確認できます。

LAN board CITIZEN SYSTEMS

HOME | STATUS | CONFIG Logout

System Status Network Status Printer Status **Service Status** Request Print

Port Number: 9210

XML Print

Service Version: 2.0

Port Number: 8080

XML Device Control

Service Version: 1.0

Port Number: 8085

LineDisplay Status: Offline

Scanner Status: Offline

Speaker Status: Offline

5. JavaScript Device Control SDK

Device Control SDK は、CITIZEN XML Device Control サービスをクライアントサイドの JavaScript で周辺機器を制御するためのライブラリです。JavaScript を使用して、Web アプリケーションから簡単に制御することが可能です。

5.1. 動作環境

本 SDK が対応する Web ブラウザーは、HTML5 に対応している必要があります。

5.2. プログラミングガイド

5.2.1. SDK ファイルの配置

Device Control SDK は JavaScript で提供しています。SDK を利用するためには、Web サーバーに「cxml-device-api.js」を配置してください。提供 SDK のソースコードに対して変更を行うと正しく動作しなくなる可能性があるため、変更を行わないでください。

5.2.2. プログラム構成

デバイスを制御するには、Web サーバーに配置した Web ページの HTML<script>タグに記述します。プログラム構成は、以下の通りです。

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Device Control SDK Sample</title>
  <script type="text/javascript" src="cxml-device-api.js"></script>
  <script type="text/javascript">
    //オブジェクトの作成(ラインディスプレイ)
    var cxml = new citizen.CXMLDisplay();
    //応答受信コールバック関数の設定
    cxml.OnReceive = function (res) {
      alert(res.ResponseCode);
    };
    //送信エラーコールバック関数の設定
    cxml.OnError = function (res) {
      alert(res.status);
    };
    //デバイス制御処理の登録
    cxml.MessageID('12345678');
    cxml.ClearDisplay(cxml.AREA_ALL);
    cxml.DisplayText('Hello, World!');
    //送信実行
    cxml.Send('http://192.168.10.100/xmldevice');
  </script>
</head>

<body>
  .
  .
</body>
</html>
```

SDK の組み込み

プログラム本体

5.2.3. オブジェクトの作成

デバイスの制御は、機能別に提供されるオブジェクトで行います。最初に使用するデバイスに合わせてオブジェクトのインスタンスを生成してください。

ラインディスプレイ制御オブジェクト: `citizen.CXMLDisplay`
バーコードスキャナー制御オブジェクト: `citizen.CXMLScanner`

5.2.4. 応答受信コールバック関数の設定

本オブジェクトの `OnReceive` プロパティにコールバック関数を設定する事により、関数の引き数の情報で制御結果を確認できます。

項目	説明
<code>ResponseCode</code>	処理結果が格納されます
<code>MessageID</code>	応答メッセージを識別のための ID が格納されます
<code>RequestID</code>	送信時に指定された <code>MessageID</code> が格納されます
<code>ErrorCode</code>	エラー発生時にエラーコードが格納されます
<code>Description</code>	エラー発生時の説明が格納されます
<code>DeviceInfo</code>	デバイスのステータス取得時に格納されます (GetDeviceInfo メソッド使用時)
<code>ScanData</code>	スキャンデータ取得時に格納されます (StartScanning , GetScanData メソッド使用時)

応答受信コールバック関数の設定例は、以下の通りです。

```
// 応答受信コールバック関数の設定
cxml.OnReceive = function (res) {
    var msg;
    if (res.ResponseCode == 'OK') {
        msg = 'Device control success!\n\n';
    }
    else {
        msg = 'Device control failure!\n\n';
        msg += ' Code: ' + res.ErrorCode + '\n';
        msg += ' Description: ' + res.Description + '\n\n';
    }
    msg += ' RequestID: ' + res.RequestID + '\n';
    alert(msg);
};
```

エラーの発生の確認

`ResponseCode` の値を確認することにより、エラーの発生の有無が確認できます。

コード	説明
OK	正常に終了した
Rejected	エラーが発生した

エラー情報の確認

エラー発生時に格納される、ErrorCode および Description の内容により、結果の要因を確認できます。
本サービスで使用するエラーコードを以下に示します。

サービス処理中に発生するエラー

コード	説明
RequestInvalid	要求情報が不正
DeviceTimeout	デバイスはタイムアウトした

デバイス処理中に発生するエラー

ENotConnect	デバイス接続失敗
EConnectNotFound	対応機種でない
EINVAL	未サポートまたは無効パラメータ
EOffline	デバイスがオフライン
ENOEXIST	データ無しエラー
EFailure	処理が実行できない
ETimeout	処理タイムアウト
EptrBadFormat	データ書式エラー
EptrTooBig	データサイズエラー

5.2.5. 送信エラーコールバック関数の設定

本オブジェクトの OnError プロパティにコールバック関数を設定する事により、関数の引き数の情報でエラー内容を確認できます。

エラー発生時のステータスが status に、レスポンスの内容が responseText に格納されます。

送信エラーコールバック関数の設定例は、以下の通りです。

```
//送信エラーコールバック関数の設定
cxml.OnError = function (res) {
  var msg = 'Send failure!\n\n';
  msg += ' status: ' + res.status + '\n';
  msg += ' responseText: ' + res.responseText + '\n';
  alert(msg);
};
```

5.2.6. デバイス制御処理

本オブジェクトでデバイス制御メソッドを呼び出す事により、デバイス制御処理を登録する事ができます。
デバイス制御メソッドの詳細は、本書の「[5.4.デバイス制御メソッド詳細](#)」を参照してください。

デバイス処理の登録例は、以下の通りです。

```
//デバイス制御処理登録
// - 要求メッセージ ID 設定 -
cxml.MessageID('12345678');
// - デバイス制御指定 -
cxml.ClearDisplay(cxml.AREA_ALL);
cxml.DisplayText('DisplayText', false);
```

5.2.7. 送信実行

本オブジェクトで XML Display Control の URL を指定して Send 関数を呼び出す事により、制御処理が開始されます。処理が終了すると、設定された応答受信コールバック関数が呼び出され制御結果を取得できます。制御結果取得の詳細は、本書の「[5.3.2. 応答受信コールバック関数の設定](#)」を参照してください。

指定する URL の書式は、以下の通りです。

http(s):// [本サービスの IP アドレス] /xmldevice

※サービスバージョン 1.0 以前では、http:// [本サービスの IP アドレス] :8085

送信実行の指定例は、以下の通りです。

```
//送信実行 (https の場合)
cxml.Send('https://192.168.10.100/xmldevice');

//送信実行 (http の場合)
cxml.Send('http://192.168.10.100/xmldevice');

//送信実行 (サービスバージョン 1.0 以前の http の場合)
cxml.Send('http://192.168.10.100:8085');
```

5.3. デバイスのステータス取得

5.3.1. デバイス情報取得メソッド(GetDeviceInfo)

デバイスのステータスを取得するには、デバイス情報取得要求を指定する GetDeviceInfo メソッドを使用します。GetDeviceInfo メソッドは、全てのオブジェクトに実装されています。

形式

GetDeviceInfo (Name, Status)

パラメータ

値	意味	設定可能範囲
Name	デバイス名	対象デバイス名 指定されなかった場合は全登録デバイスが対象
Status	ステータス情報フラグ	“true”を指定した場合にステータス情報を付加

5.3.2. 応答受信コールバック関数の設定

本オブジェクトの OnReceive プロパティにコールバック関数を設定する事により、関数の引き数の情報でデバイス情報の取得結果を確認できます。デバイスのステータス取得例は、以下の通りです。

```
//オブジェクトの作成
var cxml = new citizen.CXMLDisplay();

//応答受信コールバック関数の設定(デバイス情報確認)
cxml.OnReceive = function (res) {
    var msg = 'GetDeviceInfo failure!\n';
    if(res.ResponseCode == 'OK'){
        if((res.DeviceInfo != null) && (0 < res.DeviceInfo.length)) {
            msg = '';
            for (var i=0; i<res.DeviceInfo.length; i++){
                msg += res.DeviceInfo[i].name + ':' +
                    res.DeviceInfo[i].status + '\n';
            }
        }
    }
    alert(msg);
};

//送信エラーコールバック関数の設定
cxml.OnError = function (res) {
    alert(res.status);
};

//デバイス情報取得要求指定
cxml.GetDeviceInfo(null, "true");

//送信実行
cxml.Send('http://192.168.10.100/xmldevice');
```

デバイス情報の内容は、DeviceInfo 配列の要素の name にデバイス名が、status にデバイスの状態を示すステータスコードが格納されます。本サービスで使用するステータスコードを以下に示します。

ラインディスプレイ情報 (Name="LineDisplay")

ステータスコード	説明
Online	利用可能な状態です。
EConnectNotFound	本サービスでサポートされないデバイスが接続されています
ENotConnect	ラインディスプレイが接続されていません。
EOffline	他サービスでデバイスを利用中です。

バーコードスキャナー情報 (Name="Scanner")

ステータスコード	説明
Online	利用可能な状態です。
EConnectNotFound	本サービスでサポートされないデバイスが接続されています
ENotConnect	バーコードスキャナーが接続されていません。
EOffline	他サービスでデバイスを利用中です。
StatusScanning	デバイスからのデータ受信待ち状態です。

5.4. デバイス制御メソッド一覧

本 SDK で提供されるオブジェクトを以下に示します。

機能	オブジェクト	説明
ラインディスプレイ制御	CXMLDisplay	ラインディスプレイを制御します。
バーコードスキャナー制御	CXMLScanner	バーコードスキャナーを制御します。

5.4.1. ラインディスプレイ制御 (CXMLDisplay オブジェクト)

ラインディスプレイ制御オブジェクトで利用できる制御メソッドを以下に示します。

機能	メソッド名	説明
メッセージ ID	MessageID	発信者がメッセージを識別するために指定します。
文字列表示	DisplayText	文字を表示します。
表示クリア	ClearDisplay	表示文字を消去します。
点滅表示	BlinkDisplay	点滅表示させます。
ディスプレイ設定	SetDisplayMode	ディスプレイモードを設定します。
	SetDisplayConfig	各種設定を行います。
カーソル設定	SetCursorPosition	カーソルの位置指定をします。
	MoveCursor	カーソルを移動します。
	SetCursorType	カーソル位置を表示します。
初期化	InitializeDisplay	ディスプレイを初期化します。
文字列表示設定	SetEncode	送信する文字列のエンコードを指定します。
	SetInternationalCharacterSet	国際文字を指定します。

5.4.2. バーコードスキャナー制御 (CXMLScanner オブジェクト)

バーコードスキャナー制御オブジェクトで利用できる制御メソッドを以下に示します。

機能	メソッド名	説明
メッセージ ID	MessageID	発信者がメッセージを識別するために指定します。
連続スキャン開始	StartScanning	連続スキャンを開始します。
連続スキャン停止	StopScanning	連続スキャンを停止します。
スキャンデータ取得	GetScanData	データを取得します。
スキャンデータ取得中止	AbortScan	データ取得を中止します。

5.5. ラインディスプレイ制御メソッド詳細

5.5.1. メッセージ ID (MessageID)

形式

MessageID (ID)

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	意味	設定可能範囲
ID	要求メッセージ ID	

説明

このメソッドは、発信者がメッセージを識別するために使用します。

指定された要求メッセージ ID が制御結果の RequestID パラメータに付加されます。制御結果の詳細は本書の[「5.3.2. 応答受信コールバック関数の設定」](#)を参照してください。

使用例

```
cxml.MessageID( '12345678' );
```

5.5.2. 文字表示 (DisplayText)

形式

DisplayText (Data, ReverseFlag)

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	意味	設定可能範囲
Data	テキストデータ	
ReverseFlag	反転指定フラグ	false: 標準 true: 反転 引数省略時には、false として扱います。

説明

このメソッドは、現在のカーソル位置からテキストを表示するために使用します。

テキスト属性は、反転を指定可能です。

使用例

```
cxml.DisplayText("Hello, World!");
```


5.5.3. 表示クリア (ClearDisplay)

形式

ClearDisplay (DisplayArea)

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	意味	設定可能範囲
DisplayArea	消去領域	AREA_ALL: 全領域 AREA_CURSORLINE: カーソル行 引数省略時には、AREA_ALL として扱います。

説明

このメソッドは、表示中の文字を消去します。

使用例

```
cxml.ClearDisplay(cxml.AREA_ALL);
```

5.5.4. ディスプレイ点滅 (BlinkDisplay)

形式

BlinkDisplay (IntervalBlink)

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	意味	設定可能範囲
IntervalBlink	点滅間隔(ミリ秒)	0～

説明

このメソッドは、表示画面全体を点滅させます。

点滅間隔(ミリ秒)は、点灯と消灯の間隔を指定します。点滅間隔に、0 を指定すると、点滅は解除されます。

使用例

```
cxml.BlinkDisplay(1000)
```

5.5.5. スクリーンモード設定 (SetDisplayMode)

形式

SetDisplayMode (DisplayMode)

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	意味	設定可能範囲
DisplayMode	ディスプレイモード	OVERWRITE : 上書きモード VERTICALSCROLL : 垂直スクロールモード HORIZONTALSCROLL : 水平スクロールモード

説明

このメソッドは、以下のディスプレイのモードを設定します。

DisplayMode	概要
Overwrite	カーソル位置の文字を上書きし、カーソルを右に移動。 (カーソルが上端右端のときの入力は、カーソルを下端左端に移動、 カーソルが下端右端のときの文字入力は、カーソルを上端左端に移動)
VerticalScroll	カーソルが上端のときのカーソル上移動(または、左端での左移動)で、上端の表示行を下端にスクロール カーソルが下端のときのカーソル下移動(または、右端での右移動)で、下端の表示行を上端にスクロール
HorizontalScroll	カーソルが右端でのカーソル右移動(または、文字入力)で、現在のカーソル行に対して、左方向に文字をスクロール カーソル左端のカーソル左移動で、現在のカーソル行に対して、右方向に文字をスクロール

使用例

```
cxml.SetDisplayMode (cxml.MODE_VERTICALSCROLL)
```

5.5.6. ディスプレイ設定 (SetDisplayConfig)

形式

SetDisplayConfig (Brightness)

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	意味	設定可能範囲
Brightness	輝度 (%)	0～100

説明

このメソッドは、表示画面の輝度を変更します。

輝度は、数値が大きいほど明るくなります。0 を指定すると、画面を消灯します(表示内容は保持されます)。

設定後、表示画面全体の点滅は解除されます。

使用例

```
cxml.SetDisplayConfig(40);
```

5.5.7. カーソル設定 (SetCursorPosition)

形式

SetCursorPosition (x, y)

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	意味	設定可能範囲
x	桁位置	1～
y	行位置	1～

説明

このメソッドは、カーソル位置を設定するために使用します。

カーソル位置は、カーソルの移動座標で、桁位置と行位置を指定します。

使用例

```
cxml.SetCursorPosition(1, 2);
```

5.5.8. カーソル移動(MoveCursor)

形式

MoveCursor (dx, dy)

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	意味	設定可能範囲
dx	左右方向移動量	-128～127
dy	上下方向移動量	-128～127

説明

このメソッドは、カーソルを移動するために使用します。

現在のカーソル位置からの移動となります。カーソル移動量は、左右方向移動量(-:左方向, +:右方向)と上下方向移動量(-:上方向, +:下方向)を指定します。

使用例

```
cxml.MoveCursor(2, 0);
```

5.5.9. カーソル型の指定(SetCursorType)

形式

SetCursorType (CursorType)

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。

要素	意味	設定可能範囲
CursorType	カーソル型指定	TYPE_NONE: カーソル非表示 TYPE_UNDERLINE: カーソル表示 (省略可能要素, 省略時は TYPE_UNDERLINE)

説明

現在のカーソル位置をディスプレイに表示します。

使用例

```
cxml.SetCursorType(cxml.TYPE_UNDERLINE);
```

5.5.10. デバイス初期化 (InitializeDisplay)

形式

InitializeDisplay ()

パラメータ

ありません。

説明

デバイスを初期化します。

使用例

```
cxml.InitializeDisplay();
```

5.5.11. 文字エンコード指定 (SetEncoding)

形式

SetEncoding (Encode)

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	意味	設定可能範囲
Encode	文字エンコード	SingleByteCharacter: 8-bit Character Japanese: 日本語(katakana) SimplifiedChinese: 簡体字中国語 Korean: 韓国語 TraditionalChinese: 繁体字中国語 None: 無変換

説明

DisplayText メソッド内のテキストデータ(パラメータ Data/UTF-8 形式)をデバイスに送信するときの文字エンコードを設定します。この設定の初期値は、"SingleByteCharacter"です。

使用例

```
cxml.SetEncoding("Japanese");  
cxml.DisplayText("コンニチワ");
```

5.5.12. 国際文字の指定 (SetInternationalCharacterSet)

形式

SetInternationalCharacterSet (CharacterSet)

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	意味	設定可能範囲
CharacterSet	国際文字指定	0～16

説明

以下の国際文字セットを設定します。

CharacterSet	国際文字セット	CharacterSet	国際文字セット
0	アメリカ	9	ノルウェー
1	フランス	10	デンマークⅡ
2	ドイツ	11	スペインⅡ
3	イギリス	12	ラテンアメリカ
4	デンマークⅠ	13	韓国
5	スウェーデン	14	クロアチア
6	イタリア	15	中国
7	スペインⅠ	16	ベトナム
8	日本		

使用例

```
cxml.SetInternationalCharacterSet(8);
cxml.DisplayText("Total:¥¥1,010");
```

5.6. バーコードスキャナー制御メソッド詳細

5.6.1. メッセージ ID (MessageID)

形式

MessageID (ID)

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	意味	設定可能範囲
ID	要求メッセージ ID	

説明

このメソッドは、発信者がメッセージを識別するために使用します。

指定された要求メッセージ ID が制御結果の RequestID パラメータに付加されます。制御結果の詳細は本書の[「5.3.2.応答受信コールバック関数の設定」](#)を参照してください。

使用例

```
cxml.MessageID( '12345678' );
```

5.6.2. 連続スキャン開始 (StartScanning)

形式

StartScanning (url, callbackResult, callbackStatus, scanOnceTimeout)

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	意味	設定可能範囲
url	XML Display Control の URL	
callbackResult	開始結果を受け取るコールバック関数	
callbackStatus	状態変化を受け取るコールバック関数	
scanOnceTimeout	1 回のスキャンのタイムアウト	0～60 (省略可、省略時は 10)

説明

このメソッドは、デバイスからスキャンデータを連続して取得するために使用します。

[StopScannig](#) メソッドが実行されるまで、デバイスからデータを待ち受けます。

開始結果を受け取るコールバック関数を指定した場合、第 1 パラメータに結果を示す以下の文字列が渡されます。

文字列	説明
OK	開始処理が成功した
ENotConnect	デバイスが接続されていません
StatusScanning	スキャン中のため開始できません
ENotResponse	接続失敗

状態変化を受け取るコールバック関数を指定した場合、第 1 パラメータに結果を示す以下の文字列が渡されます。

文字列	説明
Online	オンライン
Offline	オフライン
Stoped	停止

本オブジェクトの OnReceive プロパティにコールバック関数を設定する事により、関数の引き数の情報でスキャンデータを取得できます。スキャンデータは、ScanData 配列に格納されます。

使用例

// 応答受信コールバック関数の設定(スキャンデータ取得)

```
cxml.OnReceive = function (res) {
  if(res.ResponseCode == 'OK'){
    if((res.ScanData!= null) && (0 < res.ScanData.length)) {
      msg = '';
      for (var i=0; i< res.ScanData.length; i++){
        msg += res.ScanData[i] + '\n';
      }
      alert(msg);
    }
  }
};
```

// 開始結果用コールバック関数

```
var callbackResult = function (result) {
  if (result != 'OK') {
    alert('Start error: ' + result);
  }
};
```

// 状態変化用コールバック関数

```
var callbackStatus = function (status) {
  alert('Scan status: ' + status);
};
```

// 連続スキャン開始

```
cxml.StartScanning('http://192.168.10.100/xmldevice', callbackResult, callbackStatus);
```

5.6.3. 連続スキャン停止 (StopScanning)

形式

StopScanning ()

パラメータ

ありません。

説明

このメソッドは、[StartScanning](#) メソッドで連続スキャン開始された処理を停止します。

使用例

```
cxml.StopScanning();
```


5.6.4. スキャンデータ取得 (GetScanData)

形式

GetScanData (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	意味	設定可能範囲
Timeout	タイムアウト(秒)	0～60

説明

このメソッドは、デバイスからスキャンデータを1回だけ取得するために使用します。

タイムアウトに指定した時間、デバイスからデータを待ち受けます。

本オブジェクトの OnReceive プロパティにコールバック関数を設定する事により、関数の引き数の情報でスキャンデータを取得できます。スキャンデータは、ScanData 配列に格納されます。

使用例

// 応答受信コールバック関数の設定(スキャンデータ取得)

```
cxml.OnReceive = function (res) {  
  if(res.ResponseCode == 'OK'){  
    if((res.ScanData!= null) && (0 < res.ScanData.length)) {  
      msg = '';  
      for (var i=0; i< res.ScanData.length; i++){  
        msg += res.ScanData[i] + '\n';  
      }  
      alert(msg);  
    }  
  }  
};
```

// スキャンデータ取得

```
cxml.GetScanData(10);
```

5.6.5. スキャンデータ取得中止 (AbortScan)

形式

AbortScan ()

パラメータ

ありません。

説明

このメソッドは、デバイスからのスキャンデータ取得待ち処理を中止します。

使用例

```
cxml.AbortScan();
```

5.7. SDK 設定/その他機能

SDK の設定等のその他機能について、以下を提供します。制御デバイスが利用できなくても使用できます。

関数名	説明
GetVersionCode	SDK バージョン番号を取得します。
GetVersionName	SDK バージョン文字列を取得します。

5.7.1. SDK バージョン番号の取得 (GetVersionCode)

形式

GetVersionCode ()

戻り値

バージョン番号 : Number

パラメータ

ありません。

説明

SDK のバージョン番号を数値 (Ver1.23 の場合:123) で取得します。

使用例

```
var vno = cxml.GetVersionCode();  
alert("SDK Version:" + vno/100);
```

5.7.2. SDK バージョン文字列の取得 (GetVersionName)

形式

GetVersionName ()

戻り値

バージョン文字列 : String

パラメータ

ありません。

説明

SDK のバージョン番号を文字列で取得します。

使用例

```
var vname = cxml.GetVersionName();  
alert("SDK Version:" + vname);
```

6. サンプルプログラム

JavaScript Device Control SDK のサンプルプログラムの使用方法を以下に示します。

6.1. JavaScript Device Control SDK サンプル

Web ブラウザーを起動し、サンプルプログラムを配置した URL にアクセスしてください。サンプルプログラムが実行されると、下記画面が表示されます。

画面上部の URL には、要求メッセージを送信するデバイスの URL を設定してください。
画面中央部のボタンを押すと、各サンプルプログラムが実行されます。

各サンプルプログラムの説明は、以下になります。

周辺機器	項目	説明
Display Control	Display	入力された文字をディスプレイに表示します。
	Blinking	入力された文字をディスプレイに点滅表示します。
	Reverse	入力された文字をディスプレイに反転表示します。
Scan Control	Scan start	スキャンを開始します。
	Scan stop	スキャンを停止します。
	Status	<div>Stopped</div> : 停止状態 <div>Online</div> : オンライン状態 <div>Offline</div> : オフライン状態
Device Information	Get status	デバイスのステータスを取得します。

