Plat' Home
TECHNOLOGY to serve you.

OpenBlocks IoT Family向け
データ収集ガイド

Ver.2.1.4

ぷらっとホーム株式会社

■ 商標について

・ 文中の社名、商品名等は各社の商標または登録商標である場合があります。
・ その他記載されている製品名などの固有名詞は、各社の商標または登録商標です。

■ 使用にあたって

・ 本書の内容の一部または全部を、無断で転載することはご遠慮ください。
・ 本書の内容は予告なしに変更することがあります。
・ 本書の内容については正確を期するように努めていますが、記載の誤りなどにご指摘がございましたら弊社サポート窓口へご連絡ください。
  また、弊社公開の WEB サイトにより本書の最新版をダウンロードすることが可能です。
・ 本装置の使用にあたっては、生命に関わる危険性のある分野での利用を前提とされていないことを予めご了承ください。
・ その他、本装置の運用結果における損害や逸失利益の請求につきましては、上記にかかわらずいかなる責任も負いかねますので予めご了承ください。

目次

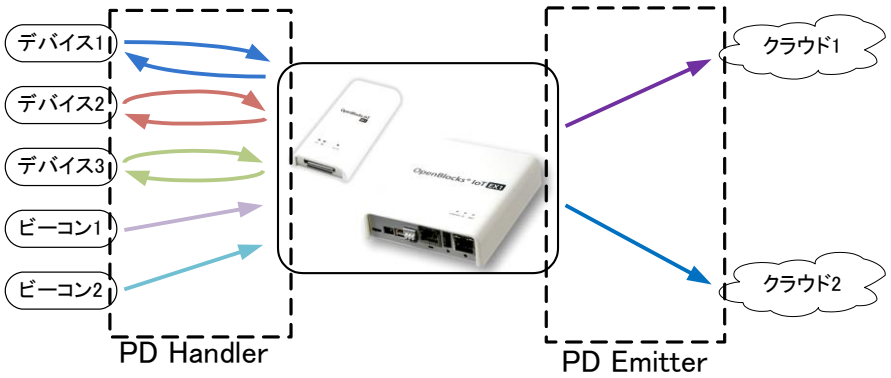
第 1 章 はじめに

本書は、OpenBlocks IoT Family にて用いているデータ収集機能について解説しています。本設定には、WEB ブラウザが使用可能なクライアント装置(PC やスマートフォン、タブレット等)が必要になります。また、WEB ユーザーインターフェース(以下、WEB UI)自体については『OpenBlocks IoT Family 向け WEB UI セットアップガイド』を参照してください。

第 2 章　データ収集機能について

OpenBlocks IoT Family 内の WEB UI のデータ収集機能はビーコン及び一部 BLE のセンサーデバイスをサポートしています。センサーデバイス等のサポート状況については、弊社 WEB ページを参照してください。

収集機能は各デバイス等からデータを取得し、各送信先のクラウド等へ情報を送信します。データを一時バッファーとして OpenBlocks IoT Family 内に保存している為、ネットワーク障害等が発生しても、再送信が行える為データを安全に送信することが出来ます。



2-1. データ収集設定

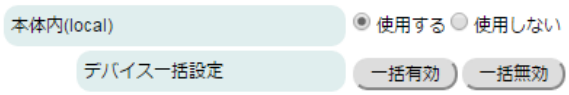
WEB UI の「サービス」→「基本」タブにてデータ収集を有効にしている場合、「収集設定」タブが表示されます。
この部分にてデータ収集の設定が行えます。
OpenBlocks IoT Family に拡張追加モジュール(EnOcean モジュール、Wi-SUN モジュール、特定小電力モジュール(FCL))を搭載している場合には基本タブにて、UART を「使用する」に設定してください。

2-1-1. 送信先設定

初期状態の送信先設定は左写真のようになっています。
ここで、ビーコンや各デバイスデータを上げる先のクラウドの設定を行います。
各項目で“使用する”を選択した場合、項目に付随する設定内容が表示されます。設定内容について、説明を行います。
送信先は”本体内(local)”を除き、最大 2 個までとなっております。

2-1-1-1. 本体内(local)

●本体内(local)



センサーデータやビーコンデータを本体内に正常に取り込めているかを確認する為の使用設定となります。

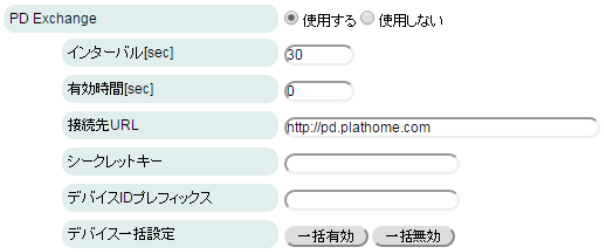
尚、本機能は PD Handler を使用している場合にのみ使用されます。

デバイス一括設定：

ビーコン及びデバイスの送信対象設定が”送信する”となっている各対象の送信先設定を一括で有効/無効を選択できます。

2-1-1-2. PD Exchange

●PD Exchange



センサーデータやビーコンデータを PD Exchange へ送信する場合の使用設定となります。

インターバル[sec]：

送信完了後～送信開始までの時間間隔を秒単位で設定します。

有効時間[sec]：

PD Emitter がデータ送信できない場合において、保持する時間を設定します。

0 を指定した場合、データ送信が完了するまで保持し続けます。

接続先 URL：

送信先の PD Exchange の URL を設定します。

シークレットキー：

接続先の PD Exchange のアカウントに対するシークレットキーを設定します。

デバイス ID プレフィックス：

接続先の PD Exchange のアカウントに対するデバイス ID プレフィックスを設定します。

デバイス一括設定：

ビーコン及びデバイスの送信対象設定が”送信する”となっている各対象の送信先設定を一括で有効/無効を選択できます。

2-1-1-3. Amazon Kinesis

●Amazon Kinesis

センサーデータやビーコンデータを Amazon Kinesis(以下、Kinesis)へ送信する場合の使用設定となります。

インターバル[sec]：

送信完了後～送信開始までの時間間隔を秒単位で設定します。

有効時間[sec]：

PD Emitter がデータ送信できない場合において、保持する時間を設定します。

0 を指定した場合、データ送信が完了するまで保持し続けます。

ドメイン名：

送信先の Kinesis のドメイン名を設定します。尚、通常変更の必要はありません。

リージョン名：

送信先の Kinesis のリージョン名を設定します。

アクセス ID：

送信先の Kinesis のアクセス ID を設定します。

アクセスキー：

送信先の Kinesis のアクセスキーを設定します。

ストリーム名：

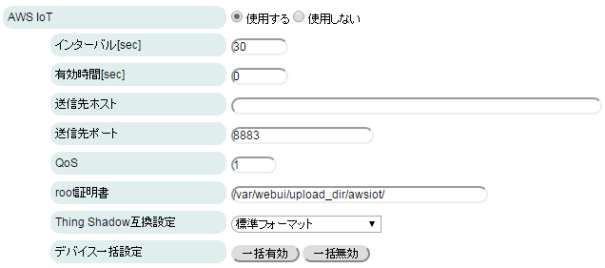
送信先の Kinesis のストリーム名を設定します。

デバイス一括設定：

ビーコン及びデバイスの送信対象設定が”送信する”となっている各対象の送信先設定を一括で有効/無効を選択できます。

2-1-1-4. AWS IoT

●AWS IoT



センサーデータやビーコンデータを AWSIoT へ送信する場合の使用設定となります。

インターバル[sec]：

送信完了後～送信開始までの時間間隔を秒単位で設定します。

有効時間[sec]：

PD Emitter がデータ送信できない場合において、保持する時間を設定します。

0 を指定した場合、データ送信が完了するまで保持し続けます。

送信先ホスト：

送信先の AWSIoT のホスト名(FQDN)を設定します。

送信先ポート：

送信先のポート番号を設定します。通常は”8883”から変更する必要はありません。

QoS：

AWSIoT へ送信する際の QoS を設定します。

“0”~”2”までが設定可能です

root 証明書：

AWSIoT へ送信する際の root 証明書を指定します。

Thing Shadow 互換設定：

ペイロードのフォーマットを標準フォーマットにするか、旧バージョンの FW(1.x 系)と同一にするかの設定を行います。

デバイス一括設定：

ビーコン及びデバイスの送信対象設定が”送信する”となっている各対象の送信先設定を一括で有効/無効を選択できます。

※root 証明書は WEB UI のシステム→ファイル管理タブにてアップロードしてください。

2-1-1-5. Waton IoT (Device)

●Watson IoT(Device) ※旧名：Bluemix

※サンプル例

センサーデータやビーコンデータを Watson IoT(Device)へ送信する場合の使用設定となります。

インターバル[sec]：

送信完了後～送信開始までの時間間隔を秒単位で設定します。

有効時間[sec]：

PD Emitter がデータ送信できない場合において、保持する時間を設定します。

0 を指定した場合、データ送信が完了するまで保持し続けます。

組織 ID：

送信先の Watson IoT(Device)の組織 ID を設定します。

quickstart を使用する場合には、“quickstart”を設定してください。

イベント ID：

送信先の Watson IoT(Device)のイベント ID を設定します。

QoS：

Watson IoT(Device)へ送信する際の QoS を設定します。

“0”~”2”までが設定可能です。

※quickstart を使用する場合には、”0”を設定する必要があります。

プロトコル：

Watson IoT(Device)へ送信する際のプロトコルを設定します。

サーバー公開証明書：

Watson IoT(Device)へ送信する際に用いるサーバー公開証明書ファイルを設定します。

デバイス一括設定：

ビーコン及びデバイスの送信対象設定が”送信する”となっている各対象の送信先設定を一括で有効/無効を選択できます。

2-1-1-6. Waton IoT (Gateway)

●Watson IoT(Gateway)

※サンプル例

センサーデータやビーコンデータを Watson IoT(Gateway)へ送信する場合の使用設定となります。

インターバル[sec]：

送信完了後～送信開始までの時間間隔を秒単位で設定します。

有効時間[sec]：

PD Emitter がデータ送信できない場合において、保持する時間を設定します。

0 を指定した場合、データ送信が完了するまで保持し続けます。

組織 ID：

送信先の Watson IoT(Gateway)の組織 ID を設定します。

イベント ID：

送信先の Watson IoT(Gateway)のイベント ID を設定します。

QoS：

Watson IoT(Gateway)へ送信する際の QoS を設定します。

“0”~”2”までが設定可能です。

※quickstart を使用する場合には、”0”を設定する必要があります。

ゲートウェイ(デバイス)タイプ：

Watson IoT(Gateway)に送信する際に用いるゲートウェイタイプを設定します。

ゲートウェイ(デバイス)ID：

Watson IoT(Gateway)に送信する際に用いるゲートウェイ ID を設定します。

パスワード：

送信先の Watson IoT(Gateway)のパスワードを設定します。

プロトコル：

Watson IoT(Gateway)へ送信する際のプロトコルを設定します。

サーバー公開証明書：

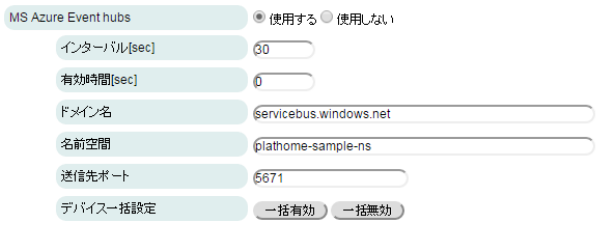
Watson IoT(Gateway)へ送信する際に用いるサーバー公開証明書ファイルを設定します。

デバイス一括設定：

ビーコン及びデバイスの送信対象設定が”送信する”となっている各対象の送信先設定を一括で有効/無効を選択できます。

2-1-1-7. Event hubs

●MS Azure Event hubs



センサーデータやビーコンデータを Event hubs へ送信する場合の使用設定となります。

インターバル[sec]：

送信完了後～送信開始までの時間間隔を秒単位で設定します。

有効時間[sec]：

PD Emitter がデータ送信できない場合において、保持する時間を設定します。

0 を指定した場合、データ送信が完了するまで保持し続けます。

ドメイン名：

送信先の Event hubs のドメイン名を設定します。

名前空間：

送信先の Event hubs の名前空間を設定します。

送信先ポート：

送信先のポート番号を設定します。通常は”5671”から変更する必要はありません。

デバイス一括設定：

ビーコン及びデバイスの送信対象設定が”送信する”となっている各対象の送信先設定を一括で有効/無効を選択できます。

2-1-1-8. IoT Hub

●MS Azure IoT Hub

センサーデータやビーコンデータを IoT Hub へ送信する場合の使用設定となります。

インターバル[sec]：

送信完了後～送信開始までの時間間隔を秒単位で設定します。

有効時間[sec]：

PD Emitter がデータ送信できない場合において、保持する時間を設定します。

0 を指定した場合、データ送信が完了するまで保持し続けます。

ドメイン名：

送信先の IoT Hub のドメイン名を設定します。

送信先ポート：

送信先のポート番号を設定します。通常は”5671”から変更する必要はありません。

IoT Hub 名：

送信先の IoT Hub 名を設定します。

デバイス一括設定：

ビーコン及びデバイスの送信対象設定が”送信する”となっている各対象の送信先設定を一括で有効/無効を選択できます。

2-1-1-9. Toami for docomo

●Toami for docomo(T4D)

センサーデータやビーコンデータを Toami for docomo へ送信する場合の使用設定となります。

インターバル[sec]：

送信完了後～送信開始までの時間間隔を秒単位で設定します。

有効時間[sec]：

PD Emitter がデータ送信できない場合において、保持する時間を設定します。

0 を指定した場合、データ送信が完了するまで保持し続けます。

接続先 URL：

送信先の Toami for docomo の URL を設定します。使用するお客様毎に URL が変更となる恐れがありますので、注意してください。

緯度：及び 経度：

本装置の緯度情報、経度情報を設定してください。システムの基本タブにて位置情報を設定している場合、同期ボタンによる自動追加等が可能です。

デバイス一括設定：

ビーコン及びデバイスの送信対象設定が”送信する”となっている各対象の送信先設定を一括で有効/無効を選択できます。

※Toami for docomo では、送信するデータの変換を行う必要があります。Toami for docomo を使用する設定にて保存ボタン選択後にキー情報変換タブが表示されますので、キー情報変換タブから設定してください。

2-1-1-10. KDDI IoT クラウドサービス STANDARD

●KDDI IoT クラウドサービス STANDARD (KDDICS)

センサーデータやビーコンデータを KDDI IoT クラウドサービス STANDARD へ送信する場合の使用設定となります。

インターバル[sec]：

送信完了後～送信開始までの時間間隔を秒単位で設定します。尚、最低値は 60 秒です。

有効時間[sec]：

PD Emitter がデータ送信できない場合において、保持する時間を設定します。

0 を指定した場合、データ送信が完了するまで保持し続けます。

ドメイン名：

送信先の KDDI IoT クラウドサービス STANDARD のドメイン名を設定します。使用するお客様のデータリンク端末基本情報をご確認の上、設定してください。

端末 ID：

KDDI IoT クラウドサービス STANDARD 側でのデータを管理する端末 ID を設定します。使用するお客様のデータリンク端末基本情報をご確認の上、設定してください。

ユーザー名：

KDDI IoT クラウドサービス STANDARD 側にて Basic 認証を行っている場合に使用するユーザー名を設定します。

パスワード：

KDDI IoT クラウドサービス STANDARD 側にて Basic 認証を行っている場合に使用するパスワードを設定します。

デバイス一括設定：

ビーコン及びデバイスの送信対象設定が”送信する”となっている各対象の送信先設定を一括で有効/無効を選択できます。

KDDI IoT クラウドサービス STANDARD に対して、HTTPS 接続の JSON 形式にてデータを POST しています。
送信するデータは弊社標準の Handler を用いている場合、JSON のデータキー及びデータ値(単位等)については、弊社 HP から OpenBlocks IoT Family データフォーマットを参照してください。
また、KDDI IoT クラウドサービス STANDARD の時間管理キーについては固定になっています。そのため、注意事項の『4-12. KDDI IoT クラウドサービス STANDARD について』を参照してください。

2-1-1-11. IoT デバイスハブ(Nifty)

●IoT デバイスハブ(Nifty)

センサーデータやビーコンデータを IoT デバイスハブ(Nifty)へ送信する場合の使用設定となります。

※tcp プロトコルの場合

※ssl プロトコルの場合

インターバル[sec]：

送信完了後～送信開始までの時間間隔を秒単位で設定します。

有効時間[sec]：

PD Emitter がデータ送信できない場合において、保持する時間を設定します。

0 を指定した場合、データ送信が完了するまで保持し続けます。

送信先ホスト：

送信先の IoT デバイスハブ(Nifty)のホストを設定します。通常は変更する必要はありません。

QoS：

送信する際の QoS を設定します。“0”~”2”までが設定可能ですが、”0”または”1”を指定してください。

プロトコル：

送信時のプロトコルを設定します。”tcp”、”ssl”から選択します。

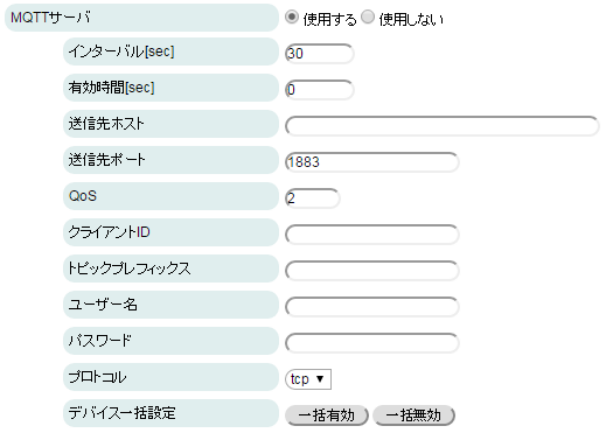
root 証明書：

“ssl”にて送信する際に使用する root 証明書を設定します。

※root 証明書は WEB UI のシステム→ファイル管理タブにてアップロードしてください。

2-1-1-12. MQTT サーバ

●MQTT サーバ



※サンプル例

センサーデータやビーコンデータを独自で構築した MQTT サーバへ送信する場合の使用設定となります。

インターバル[sec]：

送信完了後～送信開始までの時間間隔を秒単位で設定します。

有効時間[sec]：

PD Emitter がデータ送信できない場合において、保持する時間を設定します。

0 を指定した場合、データ送信が完了するまで保持し続けます。

送信先ホスト：

送信先の MQTT サーバの FQDN または IP アドレスを設定します。

送信先ポート：

送信先の MQTT サーバに接続するポート番号を指定します。通常は”1883”から変更する必要はありません。

QoS：

MQTT サーバへ送信する際の QoS を設定します。“0”~”2”までが設定可能です。

クライアント ID：

MQTT サーバへ送信する際のクライアント ID を設定します。

トピックプレフィックス：

MQTT サーバへ送信する際のトピックプレフィックスを設定します。ビーコンやセンサーの送信設定にて設定するユニーク ID (MQTT)をサフィックスとしてトピックを構成します。プレフィックスとサフィックスの間は '/' で区切られ送信されます。

ユーザー名：

送信先の MQTT サーバのユーザー名を設定します。

パスワード：

送信先の MQTT サーバのパスワードを設定します。

プロトコル：

MQTT サーバへ送信する際のプロトコルを設定します。

トラストストア：

MQTT サーバへ送信する際に用いるルート証明書ファイルを設定します。

キーストア：

MQTT サーバへ送信する際に用いるサーバ証明書ファイルを設定します。

プライベートキー：

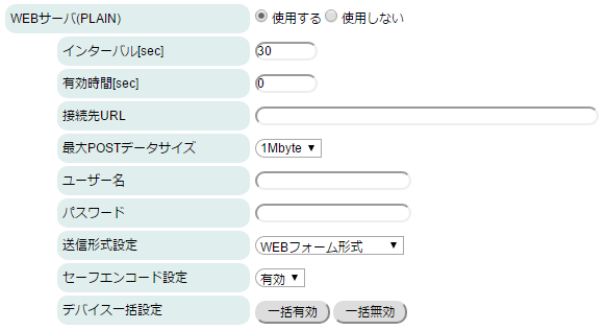
MQTT サーバへ送信する際に用いるプライベートキーファイルを設定します。

デバイス一括設定：

ビーコン及びデバイスの送信対象設定が”送信する”となっている各対象の送信先設定を一括で有効/無効を選択できます。

2-1-1-13. WEB サーバ

●WEB サーバ(PLAIN)



※サンプル例

センサーデータやビーコンデータを独自構築した WEB サーバへ送信する場合の使用設定となります。

インターバル[sec]：

送信完了後～送信開始までの時間間隔を秒単位で設定します。

有効時間[sec]：

PD Emitter がデータ送信できない場合において、保持する時間を設定します。

0 を指定した場合、データ送信が完了するまで保持し続けます。

接続先 URL：

送信先の WEB サーバの URL を設定します。

最大 POST データサイズ：

1 回の POST メソッドでの最大データサイズを選択します。1~4Mbyte の中で選択します。

ユーザー名：

WEB サーバ側にて Basic 認証を行っている場合に使用するユーザー名を設定します。

パスワード：

WEB サーバ側にて Basic 認証を行っている場合に使用するパスワードを設定します。

送信形式設定：

以下の方式となります。セーフエンコード設定に依存しますのでご注意ください。

・WEB フォーム形式

・プレーンテキスト形式

セーフエンコード設定：

URL セーフエンコードの設定を行います。

デバイス一括設定：

ビーコン及びデバイスの送信対象設定が”送信する”となっている各対象の送信先設定を一括で有効/無効を選択できます。

WEB サーバに対しては、データを POST メソッドにて送信します。送信形式設定が”WEB フォーム形式”の場合、Content-Type が "application/x-www-form-urlencoded"となります。ペイロード（送信データ本体）は、”Records” という x-www-form-urlencoded 変数に複数データをまとめて送信します。
また、送信形式設定が”プレーンテキスト形式”の場合、Content-Type が "text/plain"となります。ペイロード（送信データ本体）は、複数データ(JSON 前提)をまとめて送信します。
”プレーンテキスト形式”で送信する場合、セーフエンコード設定は「無効」にしてください。

■送信形式設定 ：WEB フォーム形式
●データ書式
Records＝[{DATA1},{DATA2},{DATA3},…{DATAn}]

●送信サンプル

```
POST / HTTP/1.0
Content-Length: 422
Content-Type: application/x-www-form-urlencoded

Records=[{"deviceId":"b0b448b81105","memo":"cc2650-1","objectTemp":20,"ambientTemp":24.84375,"humidity":47.91259765625,"temperature":25.32928466796875,"pressure":1016.37,"time":"2016-11-24T16:50:23.431+0900"},{"deviceId":"b0b448b81105","memo":"cc2650-1","objectTemp":19.75,"ambientTemp":24.875,"humidity":47.91259765625,"temperature":25.3594970703125,"pressure":1016.31,"time":"2016-11-24T16:50:26.459+0900","lux":427.68}]
```

■送信形式設定 ： プレーンテキスト形式
●データ書式
[{DATA1},{DATA2},{DATA3},…{DATAn}]

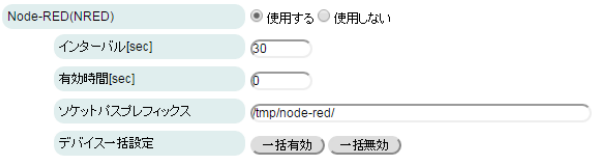
●送信サンプル

```
POST / HTTP/1.0
Content-Length: 333
Content-Type: text/plain

[{"time":"2017-04-20T11:57:13.510+09:00","deviceId":"c63021b4b969","appendixInfo":"G3E00015","rssi":-75,"latitude":35.693277,"longitude":139.740120,"ad":"xx:¥/¥/aaa"},{"time":"2017-04-20T11:57:14.111+09:00","deviceId":"e0b4865a5c44","appendixInfo":"G3E00015","rssi":-72,"latitude":35.693277,"longitude":139.740120,"ad":"xx:¥/¥/aaa"}]
```

2-1-1-14. Node-RED

●Node-RED



OpenBlocks IoT Family 内の Node-RED 構築した Unix Domain Socket に対してセンサーデータやビーコンデータを渡します。

インターバル[sec]：

送信完了後～送信開始までの時間間隔を秒単位で設定します。

有効時間[sec]：

PD Emitter がデータ送信できない場合において、保持する時間を設定します。

0 を指定した場合、データ送信が完了するまで保持し続けます。

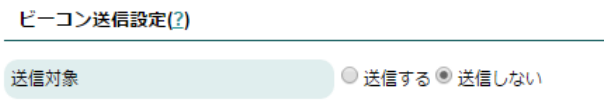
ソケットパスプレフィックス：

Node-RED に渡す為の Unix Domain Socket のファイルパスプレフィックスを設定します。

デバイス一括設定：

ビーコン及びデバイスの送信対象設定が”送信する”となっている各対象の送信先設定を一括で有効/無効を選択できます。

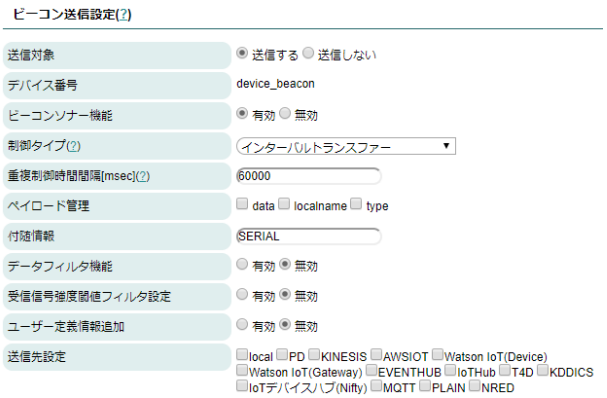
2-1-2. ビーコン送信設定



初期状態の送信先設定は左写真のようになっています。

ここで、ビーコンデータをクラウド等への送信する場合には、”送信する”を選択します。

注意）後述のデバイス情報送信設定で送信対象としているビーコンには、本項は適用されません。



”送信する”を選択した場合には、左写真のように各項目が表示されます。

デバイス番号：

OpenBlocks IoT Family の WEB UI 内で管理している番号です。変更はできません。

ビーコンソナー機能：

受信対象となっているビーコンデータを受信した際にビーコンソナーを有効にするか無効を設定します。

制御タイプ：

ビーコンデータを管理する方式を以下から選択します。各方式については後述の”ビーコン重複制御アルゴリズム”を参照してください。

・インターバルトランスファー

・エントリーポイントトランスファー

・インアウトステータストランスファー

重複制御時間間隔[ms]：

各制御タイプにて用いる制御時間を設定します。単位は msec となります。

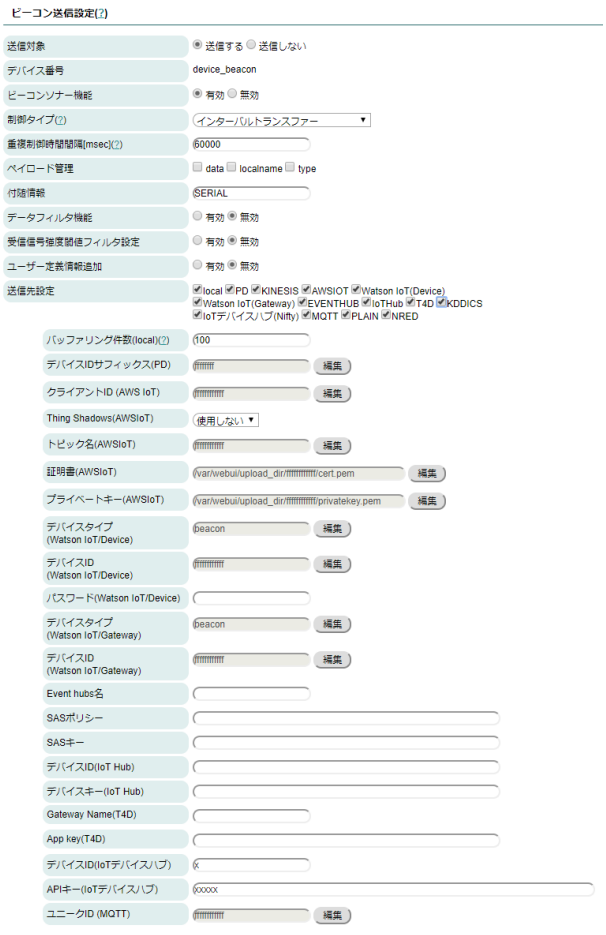
ペイロード管理：

ビーコンデータを PD Emitter へ渡す際に、ビーコンの各情報を付随させるかを選択します。

data：アドバタイズデータ(16 進数)

localname：デバイス名

type：データ種別



付随情報：

ビーコンデータを各クラウドへ送信する際に、どこの OpenBlocks IoT Family から送信されたか等の付随させる情報を設定します。

※デフォルトは本体シリアル番号です。

データフィルタ機能：(データプレフィックス)

送信対象のビーコンを選別するフィルタを設定します。データプレフィックスに 16 進文字列でフィルタ条件を入力すると、ビーコンのアドバタイズ情報を前方一致で比較し一致したもののみを送信先へ送信します。

※「追加」ボタンにて、複数登録できます。

※データフィルタを設定する場合には、本装置内(local)内のログの data を参照しデバイスをフィルタリングしてください。本装置内のログは(local)内のログについてもフィルタは適用されます。

ユーザー定義情報追加：(追加情報設定)

PD Emitter へ渡す際のデータにキー名/値の組合せで追加できます。

※「追加」ボタンにて、最大 5 個まで登録できます。

※「位置情報設定」ボタンにて、既に登録している位置情報をフォームに設定します。

受信信号強度閾値フィルタ設定：

受信対象とするビーコンの信号強度閾値フィルタを使用するか設定します。

受信信号強度閾値：

受信対象とするビーコンの信号強度を設定します。

送信先設定：

“使用する”を選択した送信先に対してチェックボックスが選択できるようになります。

チェックを付けたクラウド等に対して、送信を行います。

バッファリング件数(local)：

周囲のデバイスのアドバタイズデータを本体内に保存します。件数は最大 1 万件です。

デバイス ID サフィックス(PD)：

PD Exchange に送信する際のデバイス ID のサフィックスを設定します。

クライアント ID (AWSIoT)：

AWSIoT に送信する際のクライアント ID を設定します。Thing Shadows を使用する場合、クライアント ID が Thing Name となります。

Thing Shadows(AWSIoT)：

AWSIoT に送信する際の Thing Shadows を使用するかの設定を選択します。

トピック名(AWSIoT)：

AWSIoT に送信する際のトピックを設定します。Thing Shadows を使用する場合、トピックはクライアント ID を Thing Name として自動生成されます。

証明書(AWSIoT)：

AWSIoT に送信する際に使用するデバイスの証明書を設定します。

プライベートキー(AWSIoT)：

AWSIoT に送信する際に使用するデバイスのプライベートキーを設定します。

デバイスタイプ(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイス ID を設定します。

デバイスタイプ(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイス ID を設定します。

Event hubs 名：

Event hubs に送信する際の Event hubs 名を設定します。

SAS ポリシー：

Event hubs に送信する際の SAS ポリシーを設定します。

SAS キー：

Event hubs に送信する際の SAS キーを設定します。

デバイス ID(IoT Hub)：

IoT Hub に送信する際のデバイス ID を設定します。

デバイスキー(IoT Hub)：

IoT Hub に送信する際のデバイスキーを設定します。

Gatway Name(T4D)：

Toami for docomo に送信する際に用いる Gateway Name を設定します。

App key(T4D)：

Toami for docomo に送信する際に用いる App Key を設定します。

イベントタイプ(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いるイベントタイプを設定します。

デバイス ID(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いるデバイス ID を設定します。

API キー(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いる API キーを設定します。

ユニーク ID (MQTT)：

MQTT サーバに送信する際のユニーク ID を設定します。ユニーク ID は、トピックのサフィックスとして扱われます。トピックのプレフィックスは、MQTT サーバに設定されるトピックプレフィックスです。プレフィックスとサ

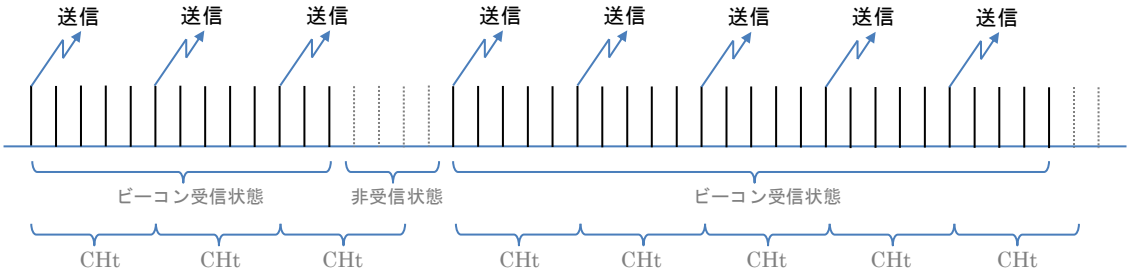
フィックスの間は '/' で区切られ送信されます。

※一部を除くクラウドに紐付く設定情報は編集ボタンにより編集可能になります。
※証明書及びプライベートキーはシステム→ファイル管理タブからアップロードしてください。
※ビーコンソナー機能を送信対象にした状態において USB スピーカー(型番：MM-SPU8BK)を接続した状態にて受信対象(データフィルタ及び受信信号強度閾値フィルタについても考慮)となっているビーコンデータを受信した場合には、スピーカーから検出音が鳴ります。

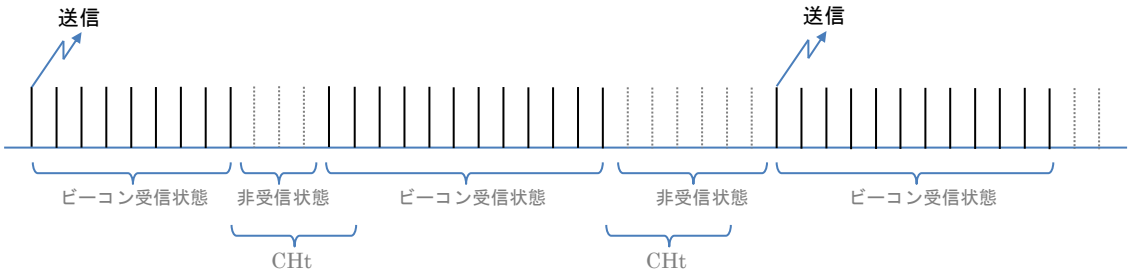
ビーコン重複制御アルゴリズム

この説明における前提条件となる設定
ビーコンの送信間隔 = 1 秒
重複制御時間間隔(CHt) = 5 秒

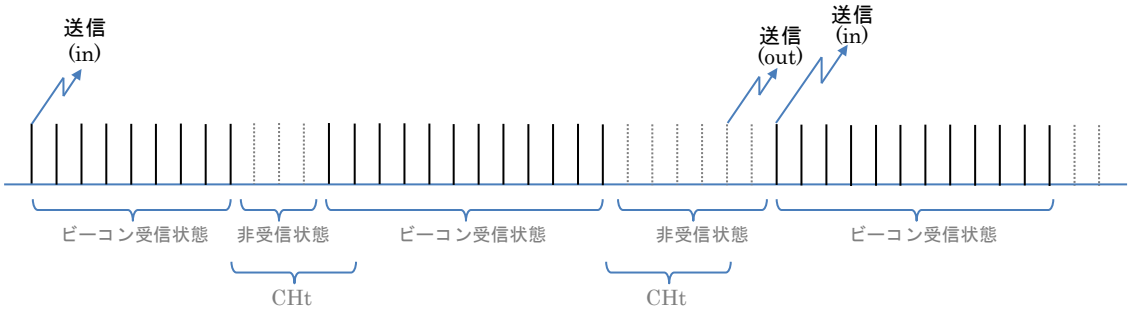
① インターバルトランスファー
ビーコンを受信している間は指定された一定間隔で送信プログラムへ。



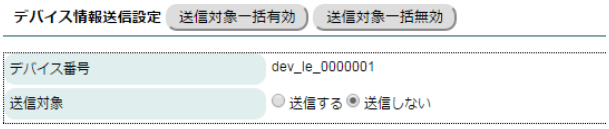
② エントリーポイントトランスファー
ビーコンが受信されたタイミングで 1 回送信プログラムへ。
（CHt 時間内の一時非受信は退場扱いしない）



③ インアウトステータストランスファー
ビーコンが入場・退場のタイミングで IN/OUT フラグ付きで送信プログラムへ。
（CHt 時間内の一時非受信は退場扱いしない）



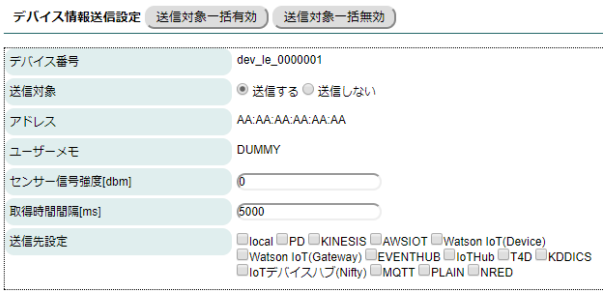
2-1-3. デバイス情報送信設定



登録済の BLE デバイスが存在している場合、初期状態では左写真のようになっています。
※BLE デバイスが 1 個登録されている場合です。

デバイス毎に送信対象項目にて”送信する”を選択すると、デバイスの送信設定の詳細を設定できます。

※送信対象一括有効、送信対象一括無効ボタンにて全ての登録済のデバイスの送信対象を制御できます。



”送信する”を選択した場合には、左写真のように各項目が表示されます。

デバイス番号：

OpenBlocks IoT Family の WEB UI 内で管理している番号です。変更はできません。

アドレス：

登録されたデバイスの BT のアドレスを表示します。

ユーザーメモ：

登録されたデバイスにて設定されたメモ情報を表示します。

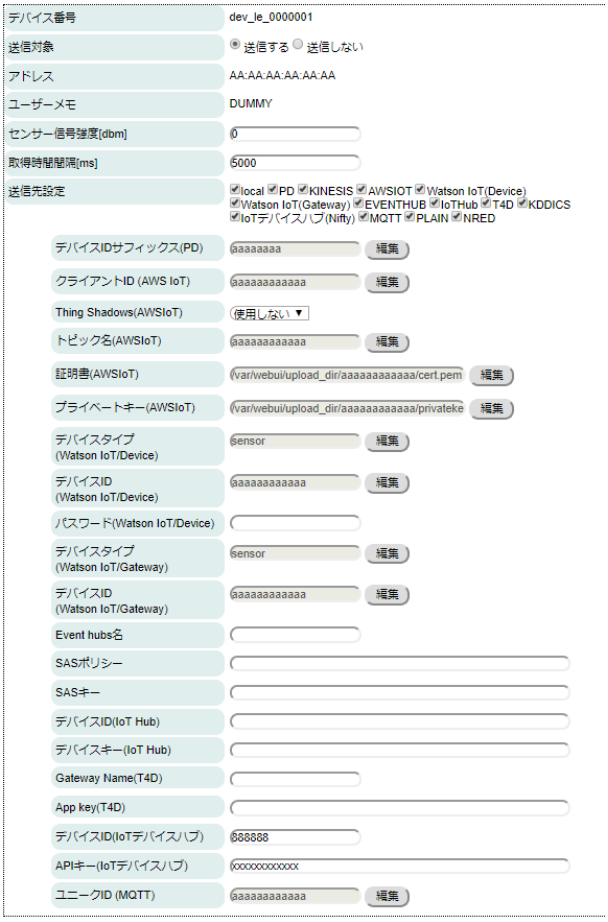
センサー信号強度[dbm]：

センサーに信号強度を設定できる機種の場合、設定したい信号強度を入力します。
設定した信号強度が無い場合、近似値またはデフォルト値が設定されます。

取得時間間隔[ms]：

センサーからデータを取得する時間間隔を数字で設定します。単位は msec です。

※”富士通コンポーネント製 BT Smart センサービーコン”と” Texas Instruments 製 SimpleLink SensorTag”のみサポートしています。



送信先設定：

“使用する”を選択した送信先に対してチェックボックスが選択できるようになります。

チェックを付けたクラウド等に対して、送信を行います。

デバイス ID サフィックス(PD)：

PD Exchange に送信する際のデバイス ID のサフィックスを設定します。

クライアント ID (AWSIoT)：

AWSIoT に送信する際のクライアント ID を設定します。Thing Shadows を使用する場合、クライアント ID が Thing Name となります。

Thing Shadows(AWSIoT)：

AWSIoT に送信する際の Thing Shadows を使用するかの設定を選択します。

トピック名(AWSIoT)：

AWSIoT に送信する際のトピックを設定します。Thing Shadows を使用する場合、トピックはクライアント ID を Thing Name として自動生成されます。

証明書(AWSIoT)：

AWSIoT に送信する際に使用するデバイスの証明書を設定します。

プライベートキー(AWSIoT)：

AWSIoT に送信する際に使用するデバイスのプライベートキーを設定します。

デバイスタイプ(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイス ID を設定します。

デバイスタイプ(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイス ID を設定します。

Event hubs 名：

Event hubs に送信する際の Event hubs 名を設定します。

SAS ポリシー：

Event hubs に送信する際の SAS ポリシーを設定します。

SAS キー：

Event hubs に送信する際の SAS キーを設定します。

デバイス ID(IoT Hub)：

IoT Hub に送信する際のデバイス ID を設定します。

デバイスキー(IoT Hub)：

IoT Hub に送信する際のデバイスキーを設定します。

Gatway Name(T4D)：

Toami for docomo に送信する際に用いる Gateway Name を設定します。

App key(T4D)：

Toami for docomo に送信する際に用いる App Key を設定します。

イベントタイプ(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いるイベントタイプを設定します。

デバイス ID(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いるデバイス ID を設定します。

API キー(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いる API キーを設定します。

ユニーク ID (MQTT) :

MQTT サーバに送信する際のユニーク ID を設定します。ユニーク ID は、トピックのサフィックスとして扱われます。トピックのプレフィックスは、MQTT サーバに設定されるトピックプレフィックスです。プレフィックスとサフィックスの間は '/' で区切られ送信されます。

※一部を除くクラウドに紐付く設定情報は編集ボタンにより編集可能になります。既存のデバイス不良等の差し替え時に以前のものと同様に扱う為に設定を同一にすることを推奨します。(不良となったデバイスは送信対象設定を“送信しない”へ変更してください。)
※証明書及びプライベートキーはシステム→ファイル管理タブからアップロードしてください。

2-1-4. PLC デバイス情報送信設定

2-1-4-1. PLC クライアント（PLC マスター）

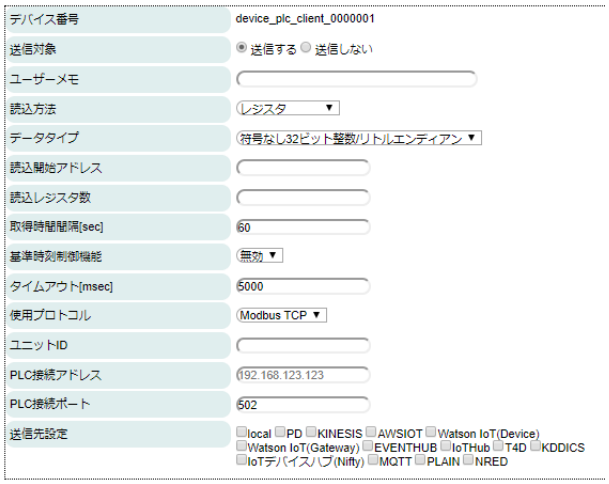
OpenBlocks IoT Family から Modbus プロトコルを用いて PLC 機器のレジスタ、コイルもしくはステータスを用いて定期的に読み込む（ポーリングを行う）場合に用います。

WEB UI の「サービス」→「基本」タブにおいて、「PD Handler PLC Client」 が「使用する」に設定されている場合、同タブの「取得 PLC 対象数」に応じた入力フォームが表示されます。

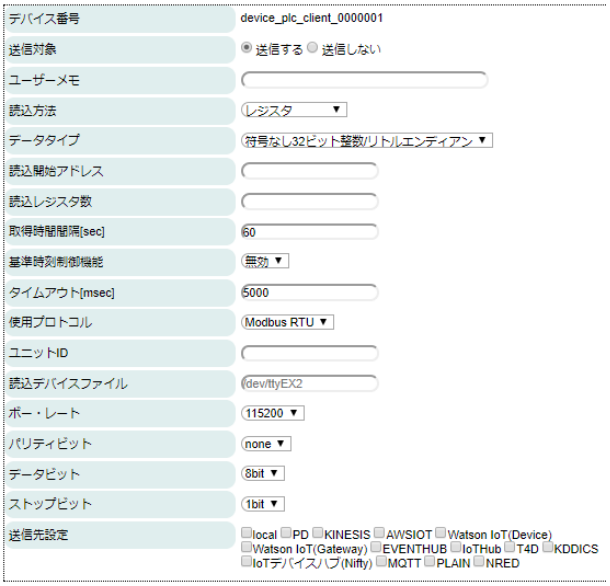
※「取得 PLC 対象」（PLC デバイス）とは、PLC 機器そのものではなく、対象となる PLC 機器への接続方法の他、データを取得するための「読込方法」や「読込開始アドレス」、「読込レジスタ数」等の設定の組み合わせを意図します。

PLC デバイス毎に送信対象項目にて”送信する”を選択すると、PLC デバイスの送信設定の詳細を設定できます。

※「使用プロトコル」として「Modbus TCP」（ネットワーク）を選択した場合の表示



※「使用プロトコル」として「Modbus RTU」（シリアル）を選択した場合の表示



”送信する”を選択した場合には、左のように各項目が表示されます。

デバイス番号：

OpenBlocks IoT Family の WEB UI 内で管理している番号です。変更はできません。

ユーザーメモ：

PLC デバイスにデータに付加する任意の文字列を設定します。データを処理する際の識別子等に利用して下さい。

読込方法：

「レジスタ」（レジスタ出力）、「入力レジスタ」（レジスタ入力）、「コイル」（ディジタル出力）、「入力ステータス」（ディジタル入力）から選択します。

「コイル」または「入力ステータス」を選んだ場合は、”0”または”1”の並びが出力されます。

データタイプ：

読込方法を「レジスタ」、「入力レジスタ」を選択した際に、出力のデータタイプを以下から選択します。

・符号なし 16 ビット整数

・符号付き 16 ビット整数

・符号なし 32 ビット整数/リトルエンディアン

・符号付き 32 ビット整数/リトルエンディアン

・符号なし 32 ビット整数/ビッグエンディアン

・符号付き 32 ビット整数/ビッグエンディアン

読込開始アドレス：

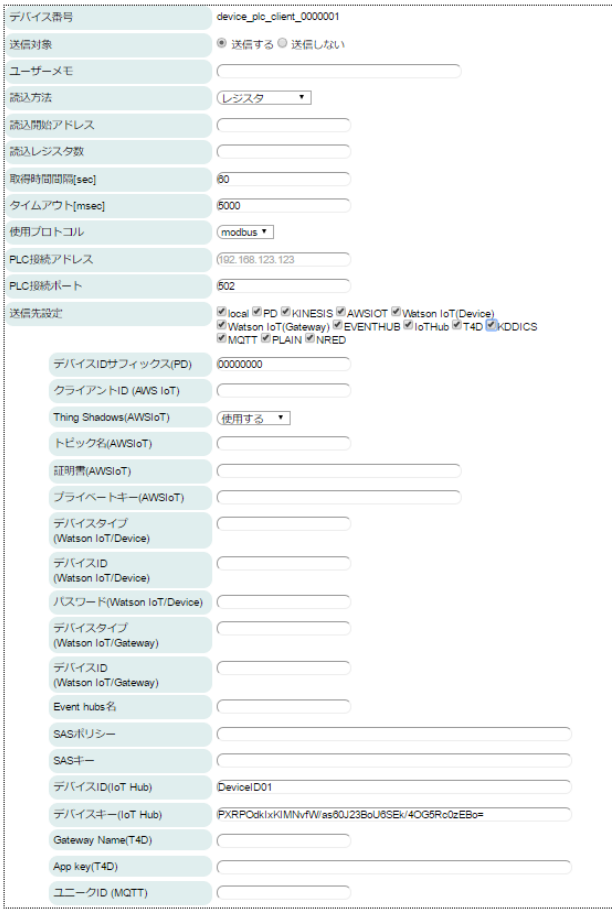
読み込みたいデータが格納されている PLC 機器上の開始アドレスを設定します。

読込レジスタ数：

「読込方法」として「コイル」または「入力ステータス」は、読み込まれるビット数と解釈されます。

「開始アドレス」に設定されるアドレスから読み込むレジスタ数もしくビット数を設定します。

※サンプル例



取得時間間隔[sec]：

PLC デバイスからデータを取得する時間間隔を数字で設定します。単位は秒です

後述の基準時刻制御を使用する場合、時間間隔は以下の値へと内部的に変更されます。

・86400[sec]使整数倍

・43200[sec]

・28800[sec]

・21600[sec]

・14400[sec]

・10800[sec]

・7200[sec]

・3600[sec]

・1800[sec]

・900[sec]

・60[sec]

基準時刻制御：

毎日定時にデータを取得する場合、本機能を有効とし基準時刻を設定しください。

基準時刻：

定時にデータを取得する際の基準時刻を設定します。HH:MM 形式となります。

タイムアウト[msec]：

PLC デバイスからデータを取得する際のタイムアウトを設定します。単位はミリ秒です。

使用プロトコル：

「Modbus TCP」、「Modbus RTU」のいずれかを選択します。

「Modbus TCP」はネットワーク、「Modbus RTU」はシリアルです。

ユニット ID：

PLC 機器の Modbus ユニット ID を設定します。ユニット ID は、1～247 または 255 の数値です。

PLC 接続アドレス(Modbus TCP)

接続する PLC 機器の IP アドレスを設定します。

PLC 接続ポート(Modbus TCP)

接続する PLC 機器の TCP ポート番号を設定します。　デフォルト値は、502 です。

読込デバイスファイル（ModbusRTU）

PLC 機器を接続するシリアルポートのデバイスファイル名を設定します。

ボー・レート（Modbus シリアル）：

PLC 機器を接続するシリアルポートのボー・レートを選択します。

パリティビット（Modbus シリアル）：

PLC 機器を接続するシリアルポートのパリティビットを選択します。

データビット（Modbus シリアル）：

PLC 機器を接続するシリアルポートのデータビット数を選択します。

ストップビット（Modbus シリアル）：

PLC 機器を接続するシリアルポートのストップビット数を選択します。

送信先設定：

“使用する”を選択した送信先に対してチェックボックスが選択できるようになります。
チェックを付けたクラウド等に対して、送信を行います。

デバイス ID サフィックス(PD)：

PD Exchange に送信する際のデバイス ID のサフィックスを設定します。

クライアント ID (AWSIoT)：

AWSIoT に送信する際のクライアント ID を設定します。Thing Shadows を使用する場合、クライアント ID が Thing Name となります。

Thing Shadows(AWSIoT)：

AWSIoT に送信する際の Thing Shadows を使用するかの設定を選択します。

トピック名(AWSIoT)：

AWSIoT に送信する際のトピックを設定します。Thing Shadows を使用する場合、トピックはクライアント ID を Thing Name として自動生成されます。

証明書(AWSIoT)：

AWSIoT に送信する際に使用するデバイスの証明書を設定します。

プライベートキー(AWSIoT)：

AWSIoT に送信する際に使用するデバイスのプライベートキーを設定します。

デバイスタイプ(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイス ID を設定します。

パスワード(Watson IoT/Device)：

Watson IoT(Device)に送信する際のパスワードを設定します。

デバイスタイプ(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイス ID を設定します。

Event hubs 名：

Event hubs に送信する際の Event hubs 名を設定します。

SAS ポリシー：

Event hubs に送信する際の SAS ポリシーを設定します。

SAS キー：

Event hubs に送信する際の SAS キーを設定します。

デバイス ID(IoT Hub)：

IoT Hub に送信する際のデバイス ID を設定します。

デバイスキー(IoT Hub)：

IoT Hub に送信する際のデバイスキーを設定します。

Gatway Name(T4D)：

Toami for docomo に送信する際に用いる Gateway Name を設定します。

App key(T4D)：

Toami for docomo に送信する際に用いる App Key を設定します。

イベントタイプ(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いるイベントタイプを設定します。

デバイス ID(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いるデバイス ID を設定します。

API キー(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いる API キーを設定します。

ユニーク ID (MQTT) :

MQTT サーバに送信する際のユニーク ID を設定します。ユニーク ID は、トピックのサフィックスとして扱われます。トピックのプレフィックスは、MQTT サーバに設定されるトピックプレフィックスです。プレフィックスとサフィックスの間は '/' で区切られ送信されます。

※一部を除くクラウドに紐付く設定情報は編集ボタンにより編集可能になります。既存のデバイス不良等の差し替え時に以前のものと同様に扱う為に設定を同一にすることを推奨します。（不良となったデバイスは送信対象設定を“送信しない”へ変更してください。）
※証明書及びプライベートキーはシステム→ファイル管理タブからアップロードしてください。

CSV ファイルを用いた「取得 PLC 対象」の拡張

/var/webui/upload_dir ディレクトリに **pd-handler-plc-client.csv** というファイル名の CSV ファイルを置くことで、WEB UI 管理にて割り当てされた 1 デバイス番号に対して複数の「取得 PLC 対象」を割り当てることが可能です。
尚、**pd-handler-plc-client.csv** ファイルは WEB UI の「システム」→「ファイル管理」タブのアップロード機能により置くことが可能です。
また、CSV ファイルの書式は、次の通りです。

デバイス番号，ユニット ID，読込方式，データタイプ，読込開始アドレス，読込レジスタ数

※行の先頭が#または/の場合、コメント行として扱います。
※CSV 内の”等での動作は保証いたしません。

パラメタ	データの型式	説明	
デバイス番号	半角英数字	WEB UI により割り振れたデバイス番号を記載します。 WEB UI に設定されていないデバイス番号は無視されます。	
ユニット ID	半角数字	PLC 機器の Modbus ユニット ID を設定します。ユニット ID は、1～247 または 255 を記載します。	
読込方式	半角英数字	読込方式として、以下のいずれかを記載します。	
		設定内容	**WEB UI 表記**
		bits	コイル
		input_bits	入力ステータス
		registers	レジスタ
		input_registers	入力レジスタ

<table>
<tr><th>パラメタ</th><th>データの型式</th><th colspan="2">説明</th></tr>
<tr><td rowspan="8">データタイプ</td><td rowspan="8">半角英数字</td><td colspan="2">データタイプとして以下を設定してください。
尚、読込方式を”bits”または”input_bits”を設定した場合、本カラムは無視されます。</td></tr>
<tr><td>設定内容</td><td>WEB UI 表記</td></tr>
<tr><td>u_int16</td><td>符号なし 16 ビット整数</td></tr>
<tr><td>int16</td><td>符号付き 16 ビット整数</td></tr>
<tr><td>u_int32lsb</td><td>符号なし 32 ビット整数/
リトルエンディアン</td></tr>
<tr><td>int32lsb</td><td>符号付き 32 ビット整数/
リトルエンディアン</td></tr>
<tr><td>u_int32msb</td><td>符号なし 32 ビット整数/
ビックエンディアン</td></tr>
<tr><td>int32msb</td><td>符号付き 32 ビット整数/
ビックエンディアン</td></tr>
<tr><td>読込開始アドレス</td><td>半角英数字</td><td colspan="2">読み込みたいデータが格納されている PLC 機器上の開始アドレスを設定します。 先頭が’0x’の場合は 16 進数と解釈されます。</td></tr>
<tr><td>読込レジスタ数</td><td>半角数字</td><td colspan="2">読み込みたいレジスタ数を記載します。</td></tr>
</table>

記載例）

```
#localname,unit_id,read_function,data_type,read_addr,read_registers
device_plc_client_0000001,15,bits,u_int16,0x130,37
device_plc_client_0000001,15,input_bits,u_int16,0x1c4,22
device_plc_client_0000001,15,registers,u_int16,0x160,3
device_plc_client_0000001,16,input_registers,u_int32lsb,0x108,1
device_plc_client_0000002,17,bits,u_int16,0x130,37
device_plc_client_0000002,18,input_bits,u_int16,0x1c4,22
device_plc_client_0000002,19,registers,int16,0x160,3
device_plc_client_0000002,20,input_registers,int32lsb,0x108,1
device_plc_client_0000003,30,bits,u_int16,0x130,37
device_plc_client_0000003,30,input_bits,u_int16,0x1c4,22
device_plc_client_0000003,31,registers,u_int16,0x160,3
device_plc_client_0000003,31,input_registers,u_int32msb,0x108,1
device_plc_client_0000004,32,bits,u_int16,0x130,37
```

```
device_plc_client_0000004,32,input_bits,u_int16,0x1c4,22
device_plc_client_0000004,33,registers,int16,0x160,3
device_plc_client_0000004,33,input_registers,int32msb,0x108,1
```

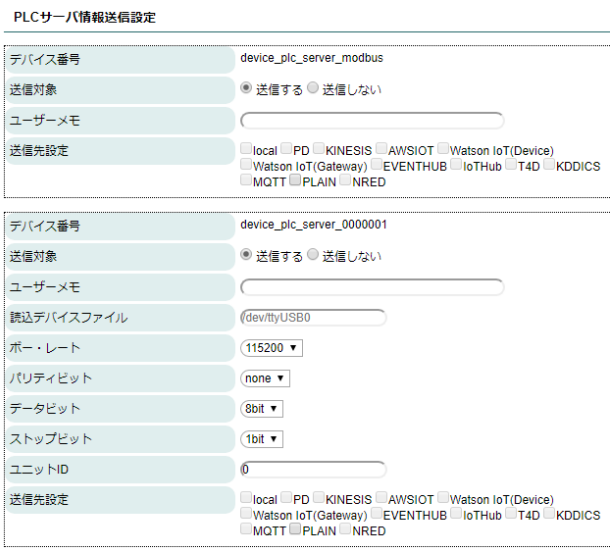
CSV ファイルに定義したデバイス番号の「取得 PLC 対象」は、WEB UI の設定内容(CSV の定義内容)は破棄され、CSV ファイルの内容が使用されます。そのため、CSV ファイルに定義したデバイス番号の設定ついては、1 デバイス番号として取得対象とする全ての「取得 PLC 対象」を記載してください。

2-1-4-2. PLC サーバ（PLC スレーブ）

Modbus プロトコルを用いて PLC 機器からレジスタ、コイルもしくはステータスを OpenBlocks IoT Family に書き込む場合に用います。
使用可能な書き込みアドレスは、レジスタ、コイルもしくはステータスのいずれも 0～2047 の範囲です。

WEB UI の「サービス」→「基本」タブにおいて、「PD Handler PLC Server」 が「使用する」に設定されている場合で、同タブの「Modbus TCP 使用設定」が有効な場合は、TCP/IP によりデータの書き込みを待ち受ける device_plc_server_modbus の入力フレームが表示されます。
TCP/IP のサーバアドレスは、WEB UI の「ネットワーク」により設定されるアドレスです。ポート番号は 502 番です。
また、同タブの「Modbus RTU 数」に応じ、シリアルポートによりデータを待ち受ける為の device_plc_server_000000n の入力フォームが表示されます。
PLC デバイス毎に送信対象項目にて”送信する”を選択すると、PLC デバイスの送信設定の詳細を設定できます。

※”送信する”を選択した場合の表示



”送信する”を選択した場合には、左のように各項目が表示されます。

ユーザーメモ：

PLC デバイスにデータに付加する任意の文字列を設定します。データを処理する際の識別子等に利用して下さい。

読込デバイスファイル（Modbus RTU）：

PLC 機器を接続するシリアルポートのデバイスファイル名を設定します。

ボー・レート（Modbus RTU）：

PLC 機器を接続するシリアルポートのボー・レートを選択します。

パリティビット（Modbus RTU）：

PLC 機器を接続するシリアルポートのパリティビットを選択します。

データビット（Modbus RTU）：

PLC 機器を接続するシリアルポートのデータビット数を選択します。

ストップビット（Modbus RTU）：

PLC 機器を接続するシリアルポートのストップビット数を選択します。

ユニット ID（Modbus RTU）：

シリアルポートに接続する PLC 機器において OpenBlocks 自体が名乗る Modbus ユニット ID を設定します。

ユニット ID は、 1～247 の整数値です。

送信先設定：

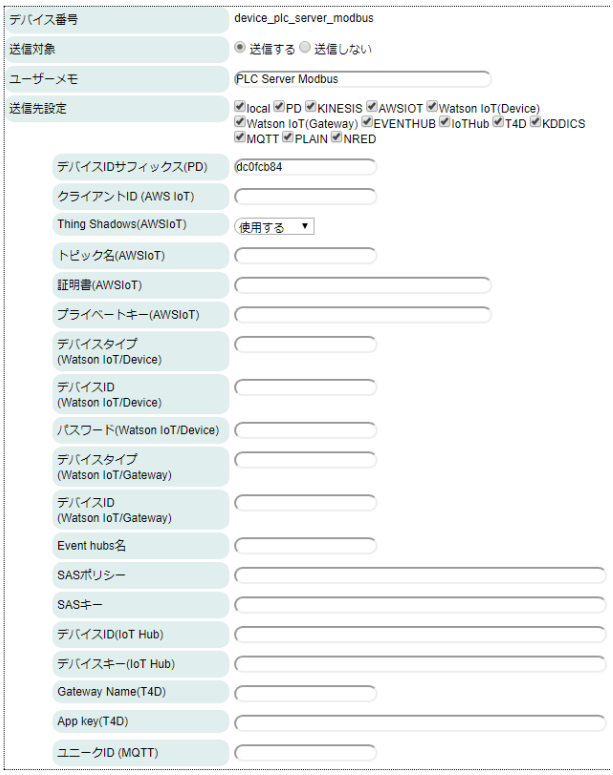
“使用する”を選択した送信先に対してチェックボックスが選択できるようになります。

チェックを付けたクラウド等に対して、送信を行います。

デバイス ID サフィックス(PD)：

PD Exchange に送信する際のデバイス ID のサフィックスを設定します。

※サンプル例



クライアント ID (AWSIoT)：

AWSIoT に送信する際のクライアント ID を設定します。Thing Shadows を使用する場合、クライアント ID が Thing Name となります。

Thing Shadows(AWSIoT)：

AWSIoT に送信する際の Thing Shadows を使用するかの設定を選択します。

トピック名(AWSIoT)：

AWSIoT に送信する際のトピックを設定します。Thing Shadows を使用する場合、トピックはクライアント ID を Thing Name として自動生成されます。

証明書(AWSIoT)：

AWSIoT に送信する際に使用するデバイスの証明書を設定します。

プライベートキー(AWSIoT)：

AWSIoT に送信する際に使用するデバイスのプライベートキーを設定します。

デバイスタイプ(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイス ID を設定します。

パスワード(Watson IoT/Device)：

Watson IoT(Device)に送信する際のパスワードを設定します。

デバイスタイプ(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイス ID を設定します。

Event hubs 名：

Event hubs に送信する際の Event hubs 名を設定します。

SAS ポリシー：

Event hubs に送信する際の SAS ポリシーを設定します。

SAS キー：

Event hubs に送信する際の SAS キーを設定します。

デバイス ID(IoT Hub)：

IoT Hub に送信する際のデバイス ID を設定します。

デバイスキー(IoT Hub)：

IoT Hub に送信する際のデバイスキーを設定します。

Gatway Name(T4D)：

Toami for docomo に送信する際に用いる Gateway Name を設定します。

App key(T4D)：

Toami for docomo に送信する際に用いる App Key を設定します。

イベントタイプ(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いるイベントタイプを設定します。

デバイス ID(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いるデバイス ID を設定します。

API キー(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いる API キーを設定します。

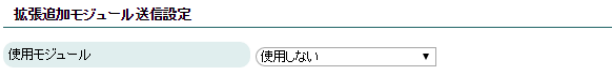
ユニーク ID (MQTT)：

MQTT サーバに送信する際のユニーク ID を設定します。ユニーク ID は、トピックのサフィックスとして扱われます。トピックのプレフィックスは、MQTT サーバに設定されるトピックプレフィックスです。プレフィックスとサフィックスの間は '/' で区切られ送信されます。

※一部を除くクラウドに紐付く設定情報は編集ボタンにより編集可能になります。既存のデバイス不良等の差し替え時に以前のものと同様に扱う為に設定を同一にすることを推奨します。（不良となったデバイスは送信対象設定を“送信しない”へ変更してください。）
※証明書及びプライベートキーはシステム→ファイル管理タブからアップロードしてください。

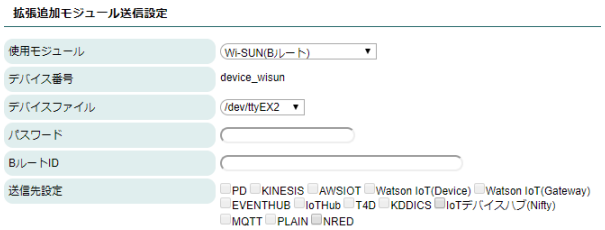
2-1-5. 拡張追加モジュール送信設定

OpenBlocks IoT Family に拡張追加モジュール(EnOcean モジュール、Wi-SUN モジュール、特定小電力モジュール(FCL)*1)を搭載している場合本項が表示されます。
本項は PD Handler UART を用いて拡張モジュールデバイスから情報を取得します。



初期状態では”使用しない”が選択されています。
データをモジュールから取得する場合には、対象モジュールを選択してください。

●Wi-SUN モジュールの場合
B ルートによる電力量の取得に対応しており、B ルートでの電力量等の取得を行う場合には使用モジュール欄にて”Wi-SUN(B ルート)”を選択します。尚、B ルート以外の通信については現在サポートしておりません。
※PD Emitter へ送信するデータの内容については、特定のキーと該当する値となります。



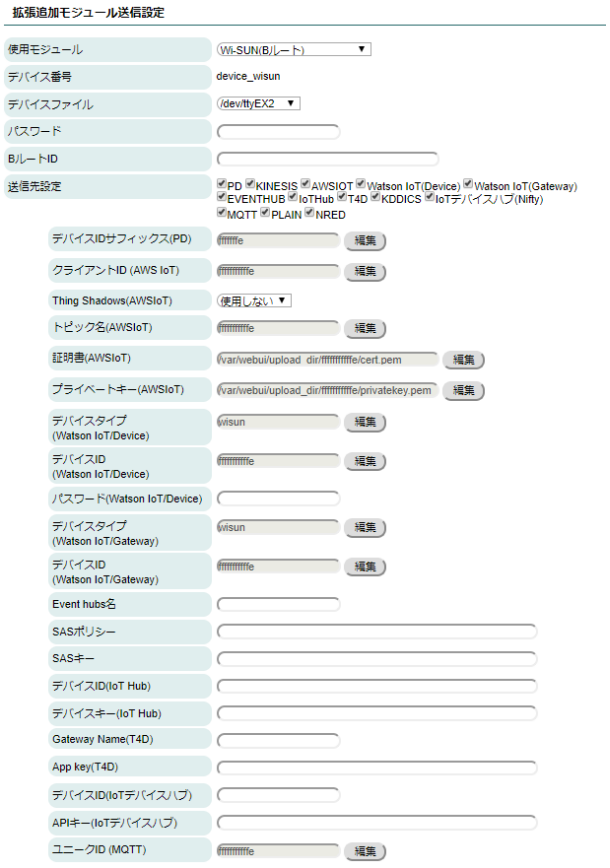
B ルートによる電力量の取得を行う場合には、電力会社から送られてくるパスワード及び B ルート ID を設定してください。

デバイスファイル：
拡張追加モジュールのデバイスファイルを選択してください。(ttyEX2 といデバイスファイルが拡張モジュールのデバイスファイルとなります。)

パスワード：
スマートメーターに接続する際のパスワードを設定してください。

B ルート ID：
スマートメーターに接続する際の B ルート ID を設定してください。
※B ルート ID は”00”から始まります。

*1 特定小電力モジュール(FCL)はβ版の実装となっております。そのため、本機能を使用する場合にはご注意ください。



送信先設定：

“使用する”を選択した送信先に対してチェックボックスが選択できるようになります。

チェックを付けたクラウド等に対して、送信を行います。

デバイス ID サフィックス(PD)：

PD Exchange に送信する際のデバイス ID のサフィックスを設定します。

クライアント ID (AWSIoT)：

AWSIoT に送信する際のクライアント ID を設定します。Thing Shadows を使用する場合、クライアント ID が Thing Name となります。

Thing Shadows(AWSIoT)：

AWSIoT に送信する際の Thing Shadows を使用するかの設定を選択します。

トピック名(AWSIoT)：

AWSIoT に送信する際のトピックを設定します。Thing Shadows を使用する場合、トピックはクライアント ID を Thing Name として自動生成されます。

証明書(AWSIoT)：

AWSIoT に送信する際に使用するデバイスの証明書を設定します。

プライベートキー(AWSIoT)：

AWSIoT に送信する際に使用するデバイスのプライベートキーを設定します。

デバイスタイプ(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイス ID を設定します。

デバイスタイプ(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイス ID を設定します。

Event hubs 名：

Event hubs に送信する際の Event hubs 名を設定します。

SAS ポリシー：

Event hubs に送信する際の SAS ポリシーを設定します。

SAS キー：

Event hubs に送信する際の SAS キーを設定します。

デバイス ID(IoT Hub)：

IoT Hub に送信する際のデバイス ID を設定します。

デバイスキー(IoT Hub)：

IoT Hub に送信する際のデバイスキーを設定します。

Gatway Name(T4D)：

Toami for docomo に送信する際に用いる Gateway Name を設定します。

App key(T4D)：

Toami for docomo に送信する際に用いる App Key を設定します。

イベントタイプ(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いるイベントタイプを設定します。

デバイス ID(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いるデバイス ID を設定します。

API キー(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いる API キーを設定します。

ユニーク ID (MQTT) :

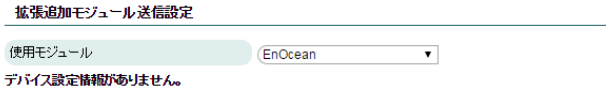
MQTT サーバに送信する際のユニーク ID を設定します。ユニーク ID は、トピックのサフィックスとして扱われます。トピックのプレフィックスは、MQTT サーバに設定されるトピックプレフィックスです。プレフィックスとサフィックスの間は '/' で区切られ送信されます。

●EnOcean モジュールの場合

EnOcean のデバイスから情報を取得する場合、使用モジュール欄にて”EnOcean”を選択します。

EnOcean デバイスのデータ収集は登録したデバイスのみ情報を取得します。登録されていないデバイスの情報は取得されませんのでご注意ください。

※PD Emitter へ送信するデータの内容については、データ送信モード及び対応 EEP に依存します。



EnOcean のデバイスが登録されていない場合、左図のように表示されます。

この場合、”EnOcean 登録”タブから EnOcean デバイスを登録してください。

EnOcean のデバイスが登録後には、左図のように表示されます。

デバイスファイル：

デバイスファイルは拡張追加モジュールのデバイスファイルを選択してください。(ttyEX2 が拡張モジュールのデバイスファイルとなります。)

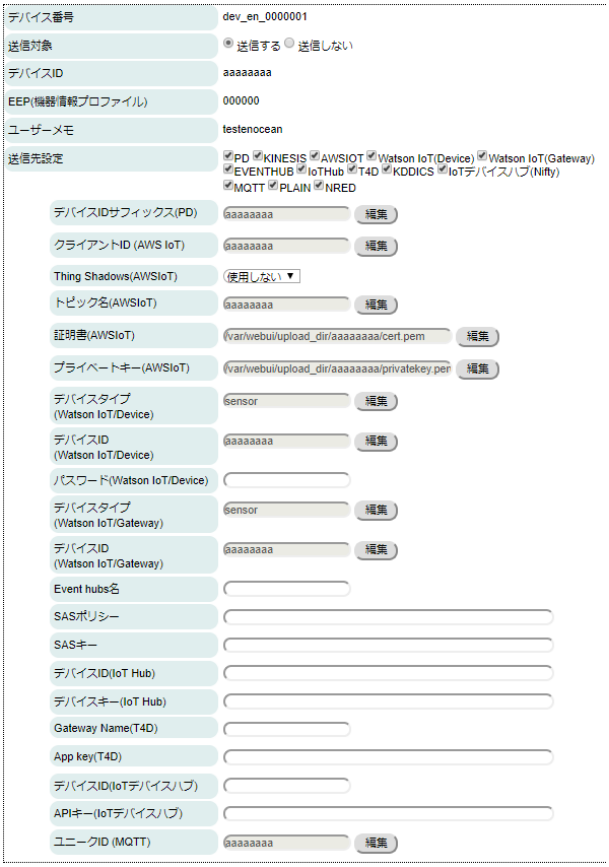
データ送信モード：

データ送信モードにて、PD Emitter へ送信するデータを設定します。データ変換モードは対応している EEP の場合は解析したデータを PD Emitter へ送信します。対応していない EEP の場合は、受信データを 16 進数文字列へ変換したデータを PD Emitter へ送信します。また、生データモードは対応 EEP を問わず、受信データを 16 進数文字列へ変換したデータを PD Emitter へ送信します。

EnOcean デバイス一括送信設定：

”送信対象一括有効”及び”送信対象一括無効”ボタンにて、全ての EnOcean デバイスの送信対象設定の一括設定が行えます。

送信対象を”送信する”を選択した場合、各項目が表示されます。



送信先設定：

“使用する”を選択した送信先に対してチェックボックスが選択できるようになります。

チェックを付けたクラウド等に対して、送信を行います。

デバイス ID サフィックス(PD)：

PD Exchange に送信する際のデバイス ID のサフィックスを設定します。

クライアント ID (AWSIoT)：

AWSIoT に送信する際のクライアント ID を設定します。Thing Shadows を使用する場合、クライアント ID が Thing Name となります。

Thing Shadows(AWSIoT)：

AWSIoT に送信する際の Thing Shadows を使用するかの設定を選択します。

トピック名(AWSIoT)：

AWSIoT に送信する際のトピックを設定します。Thing Shadows を使用する場合、トピックはクライアント ID を Thing Name として自動生成されます。

証明書(AWSIoT)：

AWSIoT に送信する際に使用するデバイスの証明書を設定します。

プライベートキー(AWSIoT)：

AWSIoT に送信する際に使用するデバイスのプライベートキーを設定します。

デバイスタイプ(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイス ID を設定します。

デバイスタイプ(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイスタイプを設定します。

デバイスタイプ(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイス ID を設定します。

Event hubs 名：

Event hubs に送信する際の Event hubs 名を設定します。

SAS ポリシー：

Event hubs に送信する際の SAS ポリシーを設定します。

SAS キー：

Event hubs に送信する際の SAS キーを設定します。

デバイス ID(IoT Hub)：

IoT Hub に送信する際のデバイス ID を設定します。

デバイスキー(IoT Hub)：

IoT Hub に送信する際のデバイスキーを設定します。

Gatway Name(T4D)：

Toami for docomo に送信する際に用いる Gateway Name を設定します。

App key(T4D)：

Toami for docomo に送信する際に用いる App Key を設定します。

イベントタイプ(IoT デバイスハブ) :

IoT デバイスハブ(Nifty)に送信する際に用いるイベントタイプを設定します。

デバイス ID(IoT デバイスハブ) :

IoT デバイスハブ(Nifty)に送信する際に用いるデバイス ID を設定します。

API キー(IoT デバイスハブ) :

IoT デバイスハブ(Nifty)に送信する際に用いる API キーを設定します。

ユニーク ID (MQTT) :

MQTT サーバに送信する際のユニーク ID を設定します。ユニーク ID は、トピックのサフィックスとして扱われます。トピックのプレフィックスは、MQTT サーバに設定されるトピックプレフィックスです。プレフィックスとサフィックスの間は '/' で区切られ送信されます。

●特定小電力モジュール(FCL)の場合

特定小電力モジュール(FCL)間同士でのデータ受信を行う場合には、使用モジュール欄にて”特定小電力モジュール(FCL)”を選択します。

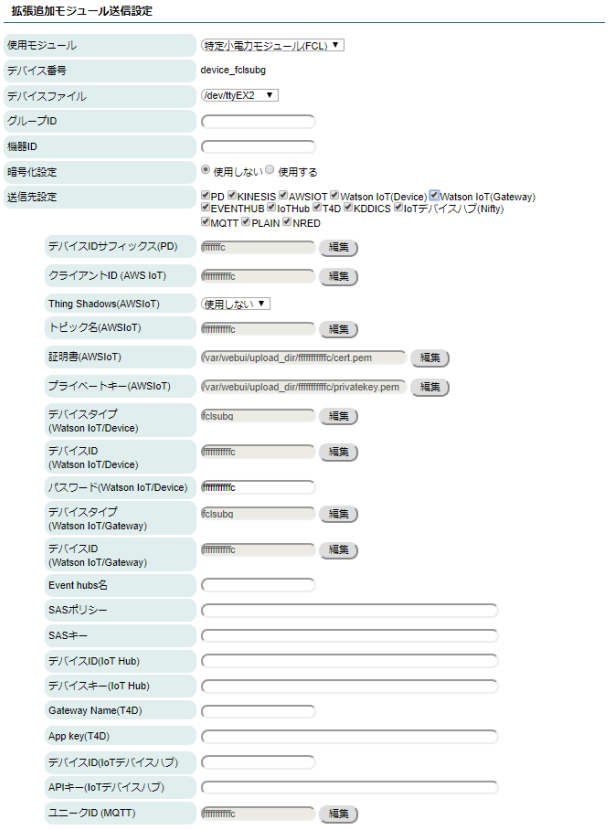
本機は特定小電力モジュール(FCL)の親機となり、特定小電力モジュール(FCL)の子機からデータを受信したデータを収集します。

※PD Emitter へ送信するデータの内容については、子機から受信データを base64 エンコードしたデータとなります。

※特定小電力モジュール(FCL)はベンダーID が固定となっておりますので同一のベンダーID のモジュールが存在する場合、対象モジュールと通信が発生する場合があります。また、評価用のモジュールはベンダーID が”0”固定となっています。

※本機能はβ版となっております。使用する場合にはご注意ください。

特定小電力モジュール(FCL)を選択した場合、左図のように表示されます。



デバイス番号：

自動的に設定されます。本項目は変更不可です。

デバイスファイル：

拡張追加モジュールのデバイスファイルを選択してください。(通常では、リストの一番下のファイルとなります。)

グループ ID：

通信を行うモジュール同士が使用する ID を入力します。入力可能値は”1”～”255”です。

機器 ID：

本モジュールの機器 ID を入力します。入力可能値は”1”～”65533”です。

暗号化設定：

通信を暗号化させるかを設定します。

暗号化鍵(32 文字)：

暗号化鍵を設定します。32 文字の 0～F の文字を入力してください。

送信先設定：

“使用する”を選択した送信先に対してチェックボックスが選択できるようになります。

チェックを付けたクラウド等に対して、送信を行います。

デバイス ID サフィックス(PD)：

PD Exchange に送信する際のデバイス ID のサフィックスを設定します。

クライアント ID (AWSIoT)：

AWSIoT に送信する際のクライアント ID を設定します。Thing Shadows を使用する場合、クライアント ID が Thing Name となります。

Thing Shadows(AWSIoT)：

AWSIoT に送信する際の Thing Shadows を使用するかの設定を選択します。

トピック名(AWSIoT)：

AWSIoT に送信する際のトピックを設定します。Thing Shadows を使用する場合、トピックはクライアント ID を Thing Name として自動生成されます。

証明書(AWSIoT)：

AWSIoT に送信する際に使用するデバイスの証明書を設定します。

プライベートキー(AWSIoT)：

AWSIoT に送信する際に使用するデバイスのプライベートキーを設定します。

デバイスタイプ(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイス ID を設定します。

デバイスタイプ(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイス ID を設定します。

Event hubs 名：

Event hubs に送信する際の Event hubs 名を設定します。

SAS ポリシー：

Event hubs に送信する際の SAS ポリシーを設定します。

SAS キー：

Event hubs に送信する際の SAS キーを設定します。

デバイス ID(IoT Hub)：

IoT Hub に送信する際のデバイス ID を設定します。

デバイスキー(IoT Hub)：

IoT Hub に送信する際のデバイスキーを設定します。

Gatway Name(T4D)：

Toami for docomo に送信する際に用いる Gateway Name を設定します。

App key(T4D)：

Toami for docomo に送信する際に用いる App Key を設定します。

イベントタイプ(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いるイベントタイプを設定します。

デバイス ID(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いるデバイス ID を設定します。

API キー(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いる API キーを設定します。

ユニーク ID (MQTT)：

MQTT サーバに送信する際のユニーク ID を設定します。ユニーク ID は、トピックのサフィックスとして扱われます。トピックのプレフィックスは、MQTT サーバに設定されるトピックプレフィックスです。プレフィックスとサフィックスの間は '/' で区切られ送信されます。

2-2. キー情報変換

一部のクラウドに対してデータを送信する場合、キー情報変換を行う必要があります。

※設定サンプル

キー情報変換：

キー変換対象クラウド：

キー変換を設定する対象のクラウドを選択します。現状では”Toami for docomo”のみサポートとなります。

テーブル追加：

変換用のテーブルの行を追加します。

エクスポート：

表示中の変換用テーブルの情報を WEB クライアントにダウンロードします。

インポート：

変換用情報を現在表示中の変換テーブルに反映します。

変換前キー：

変換元となるデータの JSON キーを設定します。

変換後キー：

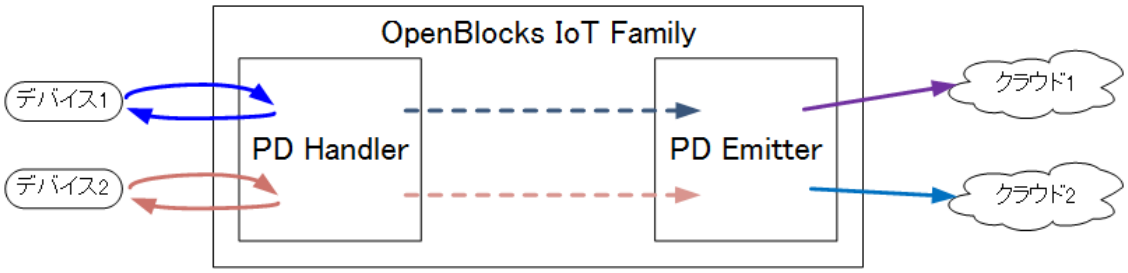
変換前キーに該当する変換後の JSON キーを設定します。

変換情報完了後に保存ボタンを押してください。

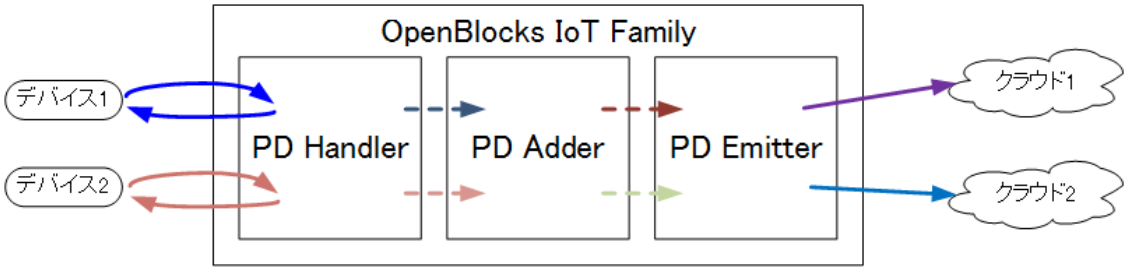
尚、変換元となる JSON キーの情報については弊社ホームページ内の本製品ページにおける各種ドキュメントを参照してください。

2-3. ペイロード付与

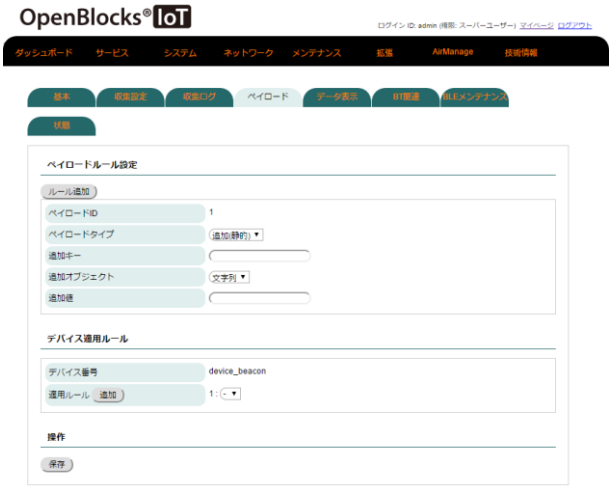
通常のデータ収集機能では以下のように各デバイス等からデータを収集する Handler から、データを取得し、クラウド等へデータを送信する PD Emitter へデータを渡しています。



WEB UI の「サービス」→「基本」タブにてペイロード付与(PD Adder)を有効にしている場合、「ペイロード」タブが表示されます。この「ペイロード」タブでは、通常のデバイスデータの他に追加データ(LTE モジュール(NTT ドコモ/KDDI)または BWA モジュールから取得した GPS 等の緯度・経度情報等)を付与することが可能です。(SIM が挿入されている必要があります)

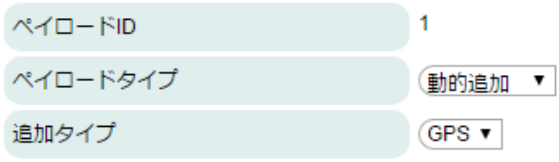


尚、ペイロード付与(PD Adder)機能にてデータ追加対象となる入力データは JSON データのみとなります。そのため Handler からの入力データが JSON データではない場合、データの付与は行われず、PD Emitter に対してデータを渡すのみとなります。



※サンプル(追加(静的))

※サンプル(動的追加)



ペイロードルール設定：

ペイロード ID：

後述のデバイス適用ルールに用いるルールの ID となります。

ペイロードタイプ：

固定値のキー及び値を追加する「追加(静的)」と、動的に変動する値を追加する「動的追加」から設定します。

追加キー： ※「追加(静的)」時のみ

追加する JSON キーを設定します。

追加オブジェクト： ※「追加(静的)」時のみ

追加する値の形式を以下の種類から選択してください。

・文字列

・整数

・小数

・真理値

追加値： ※「追加(静的)」時のみ

上記の追加オブジェクトに該当する値を設定してください。

尚、真理値の場合は”true”または”false”を設定してください。

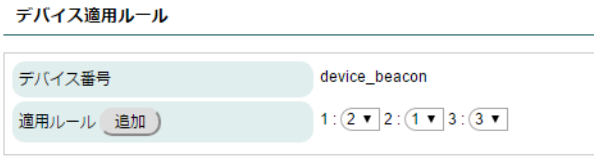
追加タイプ： ※「動的追加」時のみ

追加する値を選択します。選択可能な値については後述の表を参照してください。

尚、「ルール追加」ボタンによりペイロードルールを追加することが行えます。(最大 15 個)
また、動的追加の追加タイプ一覧は以下となります。

追加タイプ	追加キー	値内容	補足
GPS	latitude	緯度(10 進数表記) / 小数	LTE モジュール(NTT ドコモ/KDDI)または BWA モジュール使用時のみ。
	longitude	経度(10 進数表記) / 小数	

※サンプル



上記の例では、ペイロードルール”2”→ ”1”→ ”3”の順番にペイロードルールが適用されます。

デバイス適用ルール：

デバイス番号：

本製品に登録しているビーコン等のデバイスが表示されます。

適用ルール：

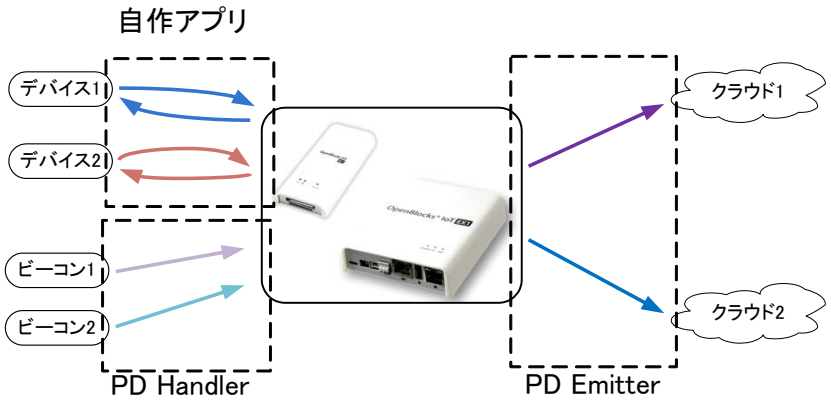
ペイロードルール設定にて作成したペイロードIDを設定します。これにより、順番通りに入力データにキー・値が追加されます。

※”-”は適用ルールなしとなります。

「追加」ボタンにより対象デバイスの適用ルールを追加することができます。(最大 5 個)
設定完了後に、「保存」ボタンを押してください。

第 3 章　デバイス連携の自作アプリ対応

OpenBlocks IoT Family 内のデータ収集機能において弊社用意のアプリケーション(PD handler)を用いず、各デバイス等からデータを取得する自作アプリを使用する場合の説明を本章にて説明を行います。
構成イメージは以下となります。



3-1. WEB UI 設定

WEB UI の「サービス」→「基本」タブにおいて、設定を行います。

通常、データ収集を行う場合、以下の表示になっているかと思います。
この状態において、デバイスからのデータ収集に自作アプリを用いる場合、「追加 Unix ドメインソケット数」の変更及びユーザーHandler 使用設定を「使用する」を選択し保存します。
尚、弊社用意の PD Handler と共存する必要が無い場合は、「データ収集」の「PD Handler BLE」を「使用しない」に設定し保存します。

※拡張モジュールを搭載した EX1 の場合には、「PD Handler UART」についても「使用しない」に設定してください。

PD Handler BLE と共存しないようにし保存ボタンを押した後ではダッシュボードを確認した場合、以下のように PD Handler BLE のプロセス状況が「停止中」となります。

これにより、PD Emitter のみ稼働している状態となります。
また、PD Emitter の設定は「サービス」→「収集設定」の状態のままとなります。

3-2. 使用 Unix ドメインソケットの送信先設定

WEB UI の「サービス」→「収集設定」タブにおいて、設定を行います。

前項目にて使用 Unix ドメインソケット数を 1 以上に設定した場合、”デバイス情報送信設定(ユーザー定義)”が表示されます。

デバイス毎に送信対象項目にて”送信する”を選択すると、デバイスの送信設定の詳細を設定できます。

送信先設定：

“使用する”を選択した送信先に対してチェックボックスが選択できるようになります。

チェックを付けたクラウド等に対して、送信を行います。

デバイス ID サフィックス(PD)：

PD Exchange に送信する際のデバイス ID のサフィックスを設定します。

クライアント ID (AWSIoT)：

AWSIoT に送信する際のクライアント ID を設定します。Thing Shadows を使用する場合、クライアント ID が Thing Name となります。

Thing Shadows(AWSIoT)：

AWSIoT に送信する際の Thing Shadows を使用するかの設定を選択します。

トピック名(AWSIoT)：

AWSIoT に送信する際のトピックを設定します。Thing Shadows を使用する場合、トピックはクライアント ID を Thing Name として自動生成されます。

証明書(AWSIoT)：

AWSIoT に送信する際に使用するデバイスの証明書を設定します。

プライベートキー(AWSIoT)：

AWSIoT に送信する際に使用するデバイスのプライベートキーを設定します。

デバイスタイプ(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Device)：

Watson IoT(Device)に送信する際のデバイス ID を設定します。

デバイスタイプ(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイスタイプを設定します。

デバイス ID(Watson IoT/Gateway)：

Watson IoT(Gateway)に送信する際のデバイス ID を設定します。

Event hubs 名：

Event hubs に送信する際の Event hubs 名を設定します。

SAS ポリシー：

Event hubs に送信する際の SAS ポリシーを設定します。

SAS キー：

Event hubs に送信する際の SAS キーを設定します。

デバイス ID(IoT Hub)：

IoT Hub に送信する際のデバイス ID を設定します。

デバイスキー(IoT Hub)：

IoT Hub に送信する際のデバイスキーを設定します。

デバイス ID(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いるデバイス ID を設定します。

API キー(IoT デバイスハブ)：

IoT デバイスハブ(Nifty)に送信する際に用いる API キーを設定します。

ユニーク ID (MQTT)：

MQTT サーバに送信する際のユニーク ID を設定します。ユニーク ID は、トピックのサフィックスとして扱われます。トピックのプレフィックスは、MQTT サーバに設定されるトピックプレフィックスです。プレフィックスとサフィックスの間は '/' で区切られ送信されます。

設定完了後に保存ボタンを押してください。

3-3. 自作アプリ向け設定

WEB UI の「サービス」→「基本」タブにおいて、設定を行います。

ユーザーHandler に関する設定を行います。

ユーザーHandler 使用設定：

ユーザー作成の Handler を使用するかを選択します。

本項目を"使用する"を選択し保存した場合、後述の起動コマンド及び停止コマンドが実行されますので、追加 Unix ドメインソケットの設定を適宜設定後に適用してください。

ユーザーHandler 起動コマンド：

ユーザーHandler 起動用のコマンドを指定します。

DAEMON 等のバックグラウンドプロセスとなる必要がありますのでご注意ください。尚、複数の Handler を用いる場合にはシェルスクリプトをラッパーとして被せて実行してください。

ユーザーHandler 停止コマンド：

ユーザーHandler 停止用のコマンドを指定します。

DAEMON 等のバックグラウンドプロセスを停止させる必要がありますのでご注意ください。

設定完了後に保存ボタンを押してください。これにより、ユーザーHandler が起動・停止されます。

3-4. 自作アプリからの PD ツールへのデータ書き込み

PD Emitter 及び PD Adder は WEB UI にて設定したデバイス番号を元に、抽象名前空間(abstract)の Unix ドメインソケットを作成します。(作成する対象は送信対象を”送信する”とし、送信先が有効でかつ local 以外が設定されているデバイスです)
この Unix ドメインソケットに対して書き込みを行った場合、書き込んだデータをクラウドへデータを送信します。
尚、対象の Unix ドメインソケットのパス規則は以下となります。
●PD Adder の Unix ドメインソケット
¥0/pd_assister/<*デバイス番号*>.sock

●PD Emitter の Unix ドメインソケット
¥0/pd_emitter_lite/<*デバイス番号*>.sock

以下は、’{“x”:1}’を各々で PD Emitter の Unix ドメインソケットに書き込みを行ったサンプルです。

コマンドラインでの書き込みサンプルは以下となります。

※<デバイス番号>：device_beacon として PD Emitter へ書き込んだ場合※1

```
# echo -n '{"x":1}' | socat stdin abstract-connect:/pd_emitter_lite/device_beacon.sock
```

PHP でのスクリプトサンプルは以下となります。

※<デバイス番号>：device_beacon として書き込んだ場合

```
<?php
    $socket = stream_socket_client("unix://¥0/pd_emitter_lite/device_beacon.sock", $errno, $errstr);
```

※1 socat コマンドはインストールされていません。そのため、”apt-get install socat”にてインストールしてください。

```
  if (!$socket) {
    echo "ERROR : " . $errno ." " . $errstr . "¥n" ;
  } else {
    fwrite($socket,'{"x":1}');
    stream_socket_shutdown($socket, STREAM_SHUT_RDWR);
  }
?>
```

Node.js でのスクリプトサンプルは以下となります。

※<デバイス番号>：device_beacon として PD Emitter へ書き込んだ場合※2

```
var absocket = require('abstract-socket');
try {
        var absclient = absocket.connect('¥0/pd_emitter_lite/device_beacon.sock', function() {
                console.log('connect ok');
        });
        absclient.write('{"x":1}');
        absclient.end();
} catch(e) {
        console.log('fail');
}
process.exit();
```

このように Unix ドメインソケットに対して、書き込みを行うことで PD Emitter のバッファーとなります。
自作アプリケーションにて、デバイス制御等を行う場合には上記のように Unix ドメインソケットへ書き込みを行ってください。
尚、ペイロード付与(PD Adder)機能を使用する場合には、PD Adder の Unix ドメインソケットに書き込みを行ってください。

※2 abstract-socket はインストールされていません。そのため、" npm install abstract-socket"にてインストールしてください。

3-5. deb パッケージによる自作アプリ連動

3-3 にて同様の設定をしていますが、deb パッケージにてインストール処理及び対応ファイルをインストールすることにより、WEB UI のフォームに入力することなく各起動制御処理と連動することが出来ます。
尚、deb パッケージの作成方法については Debian 公式ページをご確認ください。

3-5-1. インストール時処理

deb パッケージインストール時に以下のファイルの作成が必要となります。ファイルについては複数のアプリケーションにて使用する可能性があるため、内容の編集については注意してください。
このファイルに対して、WEB UI と連動させる登録アプリケーションの名称を記載します。また、1 行 1 個のアプリケーションの記載とし、空行及びアプリケーションの重複は不可です。

※対象ファイル
/etc/default/obsiot-webui-ext-handler

※ファイルサンプル(登録アプリケーション名:testhandler)

```
testhandler
```

また、ログファイルを syslog 経由で吐きだすアプリケーションの場合には、インストール時に rsyslog をリスタートしてください。(OpenBlocks IoT Family で用いている syslog サービスは rsyslog です。)

3-5-2. インストールファイル

deb パッケージ内に以下のファイルの用意する必要があります。

・/etc/default/<登録アプリケーション名>
※ファイル内容

```
bootcmd_<アプリケーション名>="<起動コマンド>"
haltcmd_<アプリケーション名>="<停止コマンド>"
statuscmd_<アプリケーション名>="<状態確認コマンド>"
```

状態確認コマンドの結果、" is running"または" RUNNING"が出力される場合に、WEB UI のダッシュボードでは稼働中として認識します。

※ファイルサンプル(登録アプリケーション：testhandler)

```
bootcmd_testhandler=" /etc/init.d/testhandler start"
haltcmd_ testhandler =" /etc/init.d/testhandler stop"
statuscmd_ testhandler ="/etc/init.d/testhandler status"
```

・アプリケーション用コンフィグファイル

WEB UI で起動制御等が実施されますが、アプリケーションに用いるコンフィグファイルは生成されません。そのため、deb パッケージにひな形となるコンフィグファイルを入れておくことを推奨します。

・ログ関連ファイル

syslog 経由にてログを出力する設定等のコンフィグファイルが必要となります。また、出力先については通常の実ストレージ領域ではなく tmpfs 領域を推奨します。WEB UI にて" /var/webui/logs"に tmpfs 領域を用意していますので、こちらに書き込んでください。尚、この領域に拡張子を".log"として用意したファイルはログ確認タブから閲覧できます。

ログを大量に吐き続けた場合、ファイルサイズが大きくなり tmpfs 領域を圧迫します。そのため、ログのローテーション設定を追加してください。tmpfs 溢れの観点からローテーション設定はファイルサイズでのトリガーを推奨とします。

第 4 章 注意事項

4-1.データ送信量及び回線速度について

ビーコンやデバイスからの情報取得量に対して、データ送信が遅い場合には、OpenBlocks IoT Family 内のバッファーに情報が溜まっていきます。この場合、データ送信部の改善を行わない場合には溜まり続けてしまう為、バッファーデータを確認しインターバルや取得時間間隔等を調整してください。
※バッファーデータは「サービス」→「状態」タブにてバッファーファイルのサイズを確認できます。

4-2.PD Emitter への書き込みデータフォーマット

PD Emitter は各クラウドへデータを送信する為、JSON データのみサポートします。
また、PD Emitter へのデータの書き込みサイズは最大 4096byte までとなります。
クラウド側でのメッセージサイズ制約が別途ありますので、使用するクラウド毎にご確認ください。

4-3.PD Emitter のバッファーサイズ

PD Emitter は送信用のバッファーとして一時溜めこみを行う為、DB にバッファーとして書き込みます。DB のサイズ上限のデフォルトは 16Mbyte です。このサイズを超えた場合、新しいデータは廃棄され、DB のサイズが 8Mbyte 以下になるまで受信は行われません。

4-4.PD Emitter のエラー時の再送信

ネットワークの通信状況によって、PD Emitter からクラウドに対しての送信が失敗することがあります。この時、連続して失敗した場合や想定外のエラー状態が発生した場合には、5 分後に再送信処理を開始します。

4-5. 自作アプリ Config について

ユーザー側にて作成した自作アプリの Config 作成機能は存在していません。ユーザー様側にて各筐体に保存する必要がありますので、ファイルアップロード機能等をご使用ください。

4-6. Toami for docomo 向けデータフォーマットについて

PD Emitter にて Toami for docomo に対して送信するデータフォーマットは JSON のみとなります。JSON 以外のフォーマットを PD Emitter に入力した場合、エラーとなります。また、エラーとなったデータは送信済みデータとして扱われますのでご注意ください。

4-7. Node-RED へのデータ経由方法について

PD Emitter から Node-RED へのデータは Unix ドメインソケット経由となります。
PD Emitter が各データにて書き込む Unix ドメインソケットのパスは以下のものとなります。

<*ソケットパスプレフィックス*>/<*デバイス番号*>.sock

Node-RED 側では、対象デバイスの「input」の「IPC」を入力として Flow に用意してください。

4-8. BLE デバイスとして追加したビーコンについて

WEB UI に BLE デバイスとして登録し送信対象として設定したセンサーやビーコンは個別に扱われます。
この場合において、特にビーコンデバイスはビーコンの送信設定との依存が無くなります。そのため、ビーコン送信設定の制御タイプ、データフィルタ等は適用されません。
また、対応しているセンサー付きビーコンの PD Emitter へ渡すデータは、解析されたセンサーデータとなります。しかし、通常のビーコンの場合は時刻/デバイス ID/メモ情報を PD Emitter へ渡します。

4-9. Toami for docomo へのデータ送信について

デバイスに設定した取得時間間隔内に再度データを受信した場合、初回のデータ以外は破棄されます。そのため、データが複数回にまたがるようなデバイス(ALPS 社製 IoT Smart Module 等)は取得時間間隔を調整してください。

4-10. PLAIN データ送信について

PD Emitter(OpenBlocks IoT のファームウェア)側からは、指定した URL の Endpoint に対して HTTP POST メソッドで送信します。
そのため、HTTP サーバ側では HTTP 200 番台のステータスコードを返す必要があります。
HTTP 200 番台のステータスコードを返却する際、HTTP ヘッダやペイロードで必要なものはありません。
尚、HTTP 200 番台以外のステータスが返された場合、PD Emitter(OpenBlocks IoT のファームウェア)側ではエラーとして扱います。
※ver.2.1.0 以下では 200 番台のステータスコードではなく、200～202 のステータスコードとなります。

4-11. Handler コンフィグユーザー設定

ver.2.1.2 以降では Handler を制御する sensor.conf のユーザー編集機能を用意しております。『サービス』→『基本』タブにて、Handler コンフィグ設定を「ユーザー定義コンフィグ」を選択し保存することで使用するものがユーザー定義のものへと切り替わります。
ファイル編集自体については、『サービス』→『編集』タブから編集を適用してください。
尚、本機能を用いた場合、サポート対象外となります。

4-12. KDDI IoT クラウドサービス STANDARD について

KDDI IoT クラウドサービス STANDARD に対してのデータ送信は“計測データフォーマット”にて送信しております。KDDI IoT クラウドサービス STANDARD では、時間軸を”datetime”キーとして扱っている為、各 Handler で用いている時刻キーと異なります。
そのため、解析オプションデータに「datetimekey="time";」を設定してください。

ぷらっとホーム株式会社
〒102-0073　東京都千代田区九段北 4-1-3　日本ビルディング九段別館 3F