Plat' Home
TECHNOLOGY to serve you.

OpenBlocks IoT VX1向け
Node-REDスターターガイド

Ver.2.0.0

ぷらっとホーム株式会社

■ 商標について

- Linux は、Linus Torvalds 氏の米国およびその他の国における商標あるいは登録商標です。
- 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。

安全上のご注意

- ご使用の前に、この「安全上のご注意」をよくお読みのうえ、正しくお使いください。
  また、お読みになったあとは大切に保管してください。
- ここに示した注意事項は、お使いになる人や、他の人への危害、財産への損害を未然に防ぐための内容を記載していますので、必ずお守りください。
- 本機の故障、誤動作または不具合などにより、通信などの機会を逸したために、お客様、または第三者が受けられた損害につきましては、当社は責任を負いかねますのであらかじめご了承ください。

表示の説明

次の表示の区分は、表示内容を守らず、誤った使用をした場合に生じる危害や損害の程度を説明しています。内容をよく理解したうえで本文をお読みください。

⚠危険	この表示は、取り扱いを誤った場合、「死亡または重傷を負う危険が切迫して生じることが想定される」内容です。
⚠警告	この表示は、取り扱いを誤った場合、「死亡または重傷を負う可能性が想定される」内容です。
⚠注意	この表示は、取り扱いを誤った場合、「軽傷を負う可能性が想定される場合および物的損害のみの発生が想定される」内容です。

絵表示の説明

次の絵表示の区分は、お守りいただく内容を説明しています。内容をよく理解したうえで本文をお読みください。

⊘禁止	禁止（してはいけないこと）を示します。
❗指示	指示に基づく行為の強制（必ず実行していただくこと）を示します。

本機、SIM カード、AC アダプタ、SD カードの取り扱いについて

⚠危険	⊘	高温になる場所（火のそば、暖房器具のそば、直射日光の当たる場所、炎天下の車内など）で使用・放置しないでください。 機器の変形・故障や内蔵電池の漏液・発熱・発火・破裂の原因となります。また、ケースの一部が熱くなり、やけどなどの原因となることがあります。
	⊘	分解・改造・ハンダ付けなどお客様による修理をしないでください。 火災・けが・感電などの事故または故障の原因となります。また、内蔵電池の漏液・発熱・破裂・発火などの原因となります。本機の改造は電波法違反となり、罰則の対象となります。
	⊘	濡らさないでください。 水などの液体が入ったときに、濡れたまま放置すると、発熱・感電・火災・けが・故障などの原因となります。使用場所、取り扱いにご注意ください。
	❗	添付された以外のACアダプタを本製品に使ったり、本製品に添付のACアダプタを他の製品に使ったりしないでください。 ACアダプタの発熱・発火・故障などの原因となります。
⚠警告	⊘	本機・ACアダプタを、加熱調理機器（電子レンジなど）・高圧容器（圧力釜など）の中に入れたり、電磁調理器（IH調理器）の上に置いたりしないでください。 内蔵電池の漏液・発熱・破裂・発火や、本機・ACアダプタの発熱・発煙・発火・故障などの原因となります。
	⊘	落としたり、投げたりして、強い衝撃を与えないでください。 内蔵電池の漏液・発熱・破裂・発火や火災・感電・故障などの原因となります。
	⊘	外部I/O端子やACアダプタ本体のプラグやUSB給電コンソールケーブル、microUSBケーブルのプラグに水などの液体や導電性異物（鉛筆の芯や金属片など）が触れないようにしてください。 また内部に入れないようにしてください。 ショートによる火災や故障などの原因となります。
	❗	プロパンガス、ガソリンなどの引火性ガスや粉塵の発生する場所（ガソリンスタンドなど）では、必ず事前に本機の電源をお切りください。 ガスに引火する恐れがあります。プロパンガス、ガソリンなど引火性ガスや粉塵の発生する場所で使用すると、爆発や火災などの原因となります。
	❗	使用中、充電中、保管時に、異音・発煙・異臭など、今までと異なることに気づいたときは、次の作業を行ってください。 1. 本機の電源を切ってください。 2. 給電用ケーブルを全て抜いて下さい。AC アダプタはアダプタ本体を持ってプラグを抜いてください。異常な状態のまま使用すると、火災や感電などの原因となります。
	❗	電池を機器に入れる場合は、＋（プラス）と－（マイナス）の向きに注意し、表示どおりに入れてください。 間違えると電池の破裂、液もれ、発火の原因になります。
⚠注意	⊘	ぐらついた台の上や傾いた所など、不安定な場所に置かないでください。 落下して、けがや故障などの原因となります。
	⊘	本機を給電機器から取り外す際は、コードを引っ張らず、プラグを持って取り外してください。 コードを引っ張るとコードが傷ついたり、端子の破損による火災や感電などの原因となります。
	⊘	ご使用環境によっては高温になる場合があります。やけどのおそれがありますので、本体底面に手を触れないようにしてください。

本機の取り扱いについて

本機の内蔵電池の種類は次のとおりです。

表示	電池の種類
CR2032/K5GK	コイン型リチウム電池

⚠ 警告	⊘	火の中に投下しないでください。 内蔵電池を漏液・破裂・発火させるなどの原因となります。
	⊘	本機内のSIMカードスロットやmicroSDカードスロットに水などの液体や金属片、燃えやすいものなどの異物を入れないでください。 火災、やけど、けが、感電の原因となります。
	❗	航空機へのご搭乗にあたり、本機の電源を切るか、機内モードに設定してください。航空機内での使用については制限があるため、各航空会社の指示に従ってください。 航空機の電子機器に悪影響を及ぼす原因となります。 なお、航空機内での使用において禁止行為をした場合、法令により罰せられることがあります。
	❗	病院での使用については、各医療機関の指示に従ってください。 使用を禁止されている場所では、本機の電源を切ってください。 電子機器や医用電気機器に悪影響を及ぼす原因となります。
	❗	高精度な制御や微弱な信号を取り扱う電子機器の近くでは、本機の電源を切ってください。電子機器が誤動作するなどの影響を与える場合があります。 ※ ご注意いただきたい電子機器の例 補聴器・植込み型心臓ペースメーカ・植込み型除細動器・その他の医用電気機器・火災報知器・自動ドア・その他の自動制御機器など。
⚠ 注意	⊘	車両電子機器に影響を与える場合は使用しないでください 本機を自動車内で使用すると、車種によりまれに車両電子機器に影響を与え、安全走行を損なう恐れがあります。
	⊘	本機に磁気カードなどを近づけないでください。 キャッシュカード・クレジットカード・テレホンカード・フロッピーディスクなどの磁気データが消えてしまうことがあります。
	⊘	指定の電池以外はご使用にならないでください。 漏液・破裂・発火の危険があります。
	❗	ご使用後の電池は充電、分解、火の中に投下するようなことはしないでください。 漏液・破裂・発火の危険があります。 また、電池を廃棄する場合は各自治体の指示に従って処分してください。

AC アダプタの取り扱いについて

⚠ 警告	⊘	使用中は、布や布団でおおったり、包んだりしないでください。 熱がこもって火災や故障などの原因となります。
	⊘	指定以外の電源・電圧で使用しないでください。 指定以外の電源・電圧で使用すると、火災や故障などの原因となります。 ACアダプタ：AC100V～240V（家庭用交流 ACコンセント専用） また、海外旅行用として、市販されている「変圧器」は使用しないでください。火災・感電・故障の原因となります。
	⊘	ACアダプタのコードが傷んだら使用しないでください。 火災、やけど、感電の原因となります。
	⊘	雷が鳴り出したら、ACアダプタには触れないでください。 感電などの原因となります。
	⊘	濡れた手でACアダプタのプラグや端子を抜き差ししないでください。 感電や故障などの原因となります。
	❗	プラグにほこりがついたときは、ACアダプタを持ってプラグをコンセントから抜き、乾いた布などで拭き取ってください。 火災の原因となります。
	❗	ACアダプタをコンセントに差し込むときは、ACアダプタのプラグや端子に導電性異物（鉛筆の芯や金属片など）が触れないように注意して、確実に差し込んでください。 感電やショートによる火災・やけど・故障などの原因となります。
	❗	本機にACアダプタを抜き差しする場合は、無理な力を加えず、水平に真っ直ぐ抜き差ししてください。 火災、やけど、けが、感電の原因となります。
	❗	長時間使用しない場合は、ACアダプタ本体を持ってプラグをコンセントから抜いてください。 感電・火災・故障の原因となります。
	❗	万一、水などの液体が入った場合は、ただちにACアダプタを持って、コンセントからプラグを抜いてください。 感電・発煙・火災の原因となります。
⚠ 注意	⊘	ACアダプタをコンセントに接続しているときは、引っ掛けるなど強い衝撃を与えないでください。 けがや故障の原因となります。
	⊘	プラグに手や指など身体の一部が触れないようにしてください。 やけど・感電・傷害・故障の原因となります。
	❗	ACアダプタをコンセントから抜くときは、コードを引っ張らず、必ずACアダプターを持ってプラグを抜いてください。 コードを引っ張るとコードが傷つき、感電や火災などの原因となります。

IoT機器を安全に利用するために

従来の人が介在するインターネット利用とは違い、IoT機器では機械同士が情報を自動でやり取りをするため、通信のセキュリティにおいて見落としがちになります。
ここではIoT機器を安全に利用するために、必要最小限考慮すべき事柄について述べます。

1. IoT機器のログイン設定においで、製品出荷時のデフォルトパスワードを必ず変更する。
2. インターネットに接続される機器は定期的にセキュリティアップデートを行う。
3. 長期停止後のIoT機器の運用開始前には、必ず始動点検を行う。
4. 通信における暗号化技術を積極的に導入する。
5. ハードウェアが本来接続された本物かを判断できる認証技術をなるべく導入する。
6. その他、総務省が発行する「IoTセキュリティガイドライン」を参考にする。

Bluetooth®／Wi-Fi（無線LAN）ご使用上の注意

● 本機の Bluetooth® 機能および Wi-Fi（無線 LAN）機能は、2.4GHz 帯の周波数を使用します。

［現品表示］
Bluetooth® 機能：2.4 FH8
本機は 2.4GHz 帯を使用します。FH8 は、変調方式として FH-SS 変調方式を採用し、与干渉距離は約 80m 以下です。
Wi-Fi（無線 LAN）機能：2.4DS/OF4
本機は 2.4GHz 帯を使用します。変調方式として DS-SS 方式および OFDM 方式を採用しています。与干渉距離は約 40m 以下です。

2400MHz ～ 2483.5MHz の全帯域を使用し、かつ移動体識別装置の帯域を回避可能です。

● 本製品の使用周波数帯では、電子レンジ等の産業・科学・医療用機器のほか工場の製造ライン等で使用されている移動体識別用の構内無線局（免許を要する無線局）および特定小電力無線局（免許を要しない無線局）が運用されています。
(1). 本製品を使用する前に、近くで移動体識別用の構内無線局及び特定小電力無線局が運用されていないことを確認してください。
(2). 万一、本製品から移動体識別用の構内無線局に対して電波干渉の事例が発生した場合には、速やかに電波の発射を停止した上、下記の連絡先にご連絡頂き、混信回避のための処置等（例えば、パーティションの設置など）についてご相談してください。
(3). その他、本製品から移動体識別用の特定小電力無線局に対して電波干渉の事例が発生した場合など何かお困りのことが起きたときは、次の連絡先へお問い合わせください。
連絡先：ぷらっとホーム株式会社　TEL：03-5213-4372　E-Mail：support@plathome.co.jp

本機は5GHzの周波数帯においてW52のチャンネルを使用できます。W52は、電波法により屋外での使用が禁じられています。

本機の Bluetooth® ／ Wi-Fi（無線 LAN）機能は日本国内規格に準拠し、認定を取得しています。一部の国／地域では Bluetooth® ／ Wi-Fi（無線 LAN）機能の使用が制限されることがあります。海外でご利用になる場合は、その国／地域の法規制などの条件をご確認ください。

その他のご注意

・この装置は、クラス A 情報技術装置です。この装置を家庭環境で使用すると電波妨害を引き起こすことがあります。この場合には使用者が適切な対策を講ずるよう要求されることがあります。VCCI-A
・本製品は、国内での使用を前提に作られています。
海外での使用につきましては、お客様の責任で行っていただくようお願いいたします。
・DC Wide 入力をご使用いただく場合、下記及び同等の外付けノイズフィルターの接続が必要です。
推奨ノイズフィルタ：NAC-04-472(COSEL)
・本製品に搭載されている記憶媒体は eMMC で、書き込み回数に制限が設けられた有寿命部品です。修理の際、書き込み上限に達していることが確認された場合には保証期間内であっても有償修理となります。
・周囲温度が 40℃を超える環境に本製品を設置する場合は、添付の放熱・設置ブラケットを取り付けてご使用ください。

目次

第 1 章 はじめに

本書は、OpenBlocks IoT VX1 に搭載されている Node-RED の使用方法を解説しています。搭載している Node-RED はデータ収集機能にて送信先の候補として用意しており、エッジコンピューティングの実装や対応していないクラウドへの対応を想定しております。

第 2 章 Node-RED 事前準備

2-1. Node-RED の使用設定について

WEB UI の「拡張」→「node red」タブから使用設定を実施してください。

Node-RED を使用する場合には、“使用する”に設定し保存ボタンを押してください。

また、ログイン認証を使用する場合には “使用する” を選択し、ユーザー名及びパスワードを設定し保存ボタンを押してください。

2-2. Node-RED のブラウザフィルタについて

WEB UI の「システム」→「フィルター」タブにて Node-RED のフィルターを開放してください。

デフォルトでは Node-RED のブラウザアクセスはできないようにフィルターが適用されています。

“有効”に設定し、保存ボタンを押してください。

尚、セキュリティの観点上、Node-RED の設定完了後はフィルターを閉鎖してください。

2-3. パケットフィルタについて

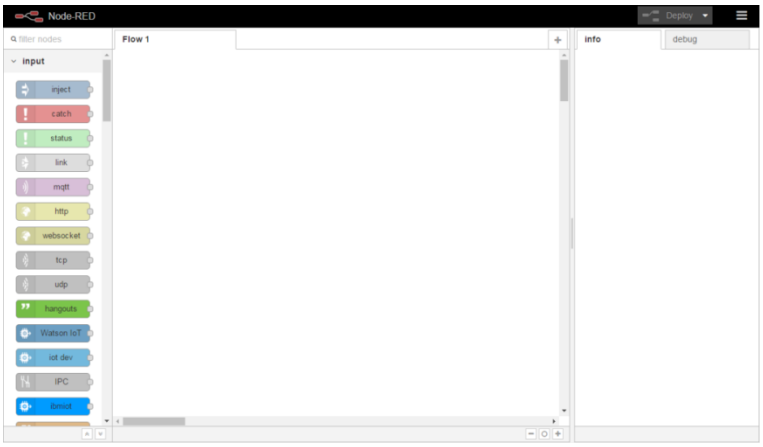
Openblocks IoT ファミリーの入力方向のパケットフィルタは、WebUI へのアクセスや時刻同期等、システムの動作に必要となるポートを除き開放されておりません。
TCP Input node 等、リモートからの接続を待ち受けるノードを使用する場合、必要に応じて別途パケットフィルタを開放する必要があります。
パケットフィルタの操作は、WEB UI の「拡張」→「スクリプト編集」タブから、iptable を操作するスクリプトを設定してください。

第 3 章 Node-RED の簡易説明

Node-RED はデフォルトで 1880 番ポートを用いてブラウザアクセス行います。
そのため、デフォルトでの Wi-Fi 経由での Node-RED へのアクセスする為の URL は以下となります。

http://192.168.254.254:1880/

アクセスした場合、以下のような画面が初期状態では表示されます。(ログイン認証設定をしていない場合)



また、本製品向けにデフォルトで用意している入力・出力・処理等のノードは以下となります。

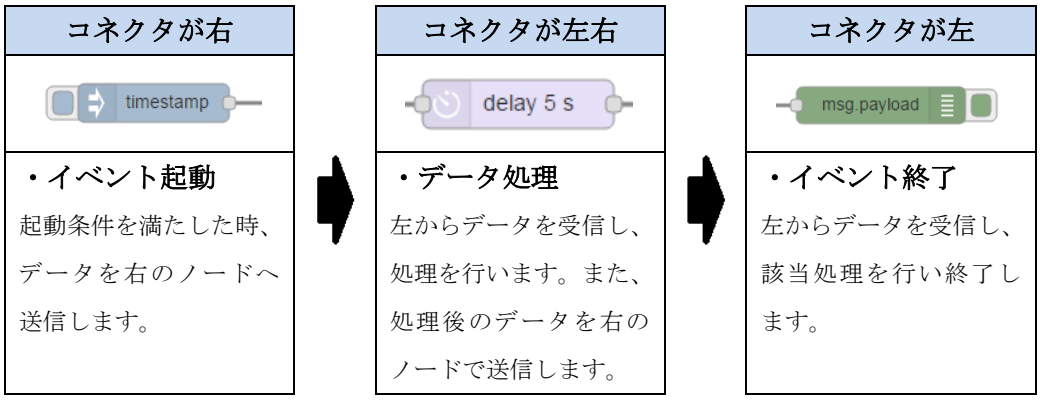
3-1. Node-RED 画面構成

Node-RED の画面は以下のように構成されています。

#	項目	説明
1	シート	処理フローを記述するワークスペースです。
2	デプロイ	デプロイボタンをクリックすることでシートに記述した処理フローを有効化します。
3	フロー	ノードを配置し結線することでデータの流れ（処理フロー）を定義します。
4	ノードパレット	処理フローの構成に用いられるノードの一覧です。
5	表示切替	ノード情報・デバック情報の表示を切り替えます。
6	ノード情報	ノード情報、又はデバック情報が表示されます。

3-2. ノード種類

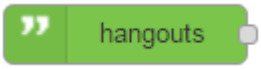
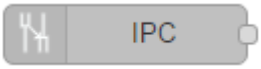
Node-RED では大きく分けて以下のようなコネクタ配置のノードがあります。



上記のように処理が行われるため、データは左から右へと処理が行われます。

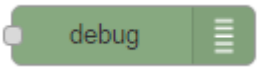
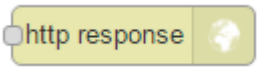
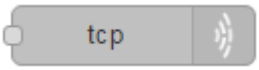
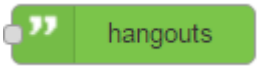
3-2. Input ノード

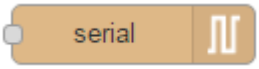
inject	ノードの左部のボタンを押すことでノードに設定された timestamp 等を入力データもしくはイベントとします。
catch	同じシート上のノードで発生したエラーを入力データもしくはイベントとします。
status	同じシート上のノードのステータスを入力データもしくはイベントとします。
link	いずれかの link output node の出力を入力データもしくはイベントとします。
mqtt	MQTT broker へ subscrib し、publish message を待ち受け、入力データもしくはイベントとします。
http	HTTP リクエストを待ち受け、入力データもしくはイベントとします。
websocket	Websocket による接続を待ち受け、入力データもしくはイベントとします。
tcp	TCP による接続を待ち受け、入力データもしくはイベントとします。
udp	UDP による接続を待ち受け、入力データもしくはイベントとします。

	Google hangouts からのメッセージを待ち受け、入力データもしくはイベントとします。
	Watoson IoT からの device command を待ち受け、入力データもしくはイベントとします。*1
	Watoson IoT からの device command を待ち受け、入力データもしくはイベントとします。*1
	UNIX ドメインソケットからの入力を待ち受け、入力データもしくはイベントとします。
	Watoson IoT からの device command を待ち受け、入力データもしくはイベントとします。*1
	シリアルインタフェースからの入力を待ち受け、入力データもしくはイベントとします。

*1：Input ノード Watson IoT と iot dev, ibmiot は、同等の機能を提供します。

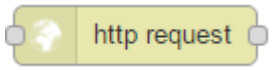
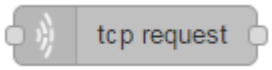
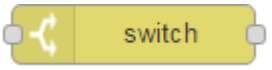
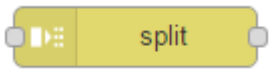
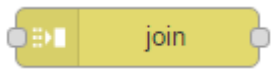
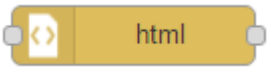
3-3. Output ノード

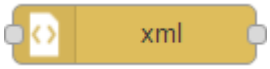
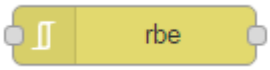
	入力データをデバック情報として表示します。
	入力データをいずれかの Input link node へ出力します。
	入力データを MQTT broker へ publish します。
	入力データを http input node への入力に対する応答として出力します。
	入力データを Websocket サーバへ出力します。
	入力データを TCP サーバに出力します。
	入力データを UDP サーバに出力します。
	入力データを Google hangouts へ出力します。
	入力データを Watson IoT へ出力します。*1
	入力データを Watson IoT へ出力します。*1

	入力データを Watson IoT へ出力します。*1
	入力データをシリアルインタフェースへ出力します。

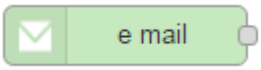
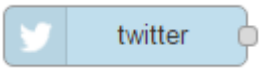
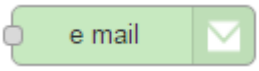
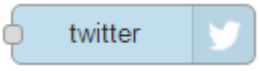
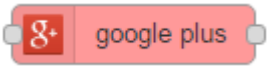
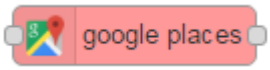
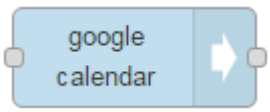
*1：Output ノード Watson IoT と iot dev, ibmiot は、同等の機能を提供します。

3-4. function ノード

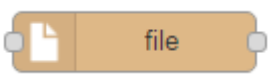
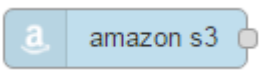
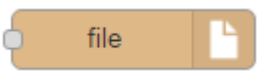
	入力データを JavaScript で処理し出力します。
	入力データを整形して出力します。
	入力データを設定された時間後に出力します。
	入力データに対し、タイムアウトを設け２つのメッセージを出力します。
	フローにコメントを付けます。
	入力データに対し、設定された URL に対し http リクエストを行い、その応答を出力します。
	入力データに対し、設定されたサーバに対し TCP 接続を行い、その応答を出力します。
	入力データを設定された分岐条件に応じて異なるノードに出力します。
	入力データの属性を設定・変更・削除もしくは移動して出力します。
	入力データのスケールを変更して出力します。
	入力データを設定される文字で区分して出力します。
	入力データを結合して出力します。
	CSV 書式のデータと JavaScript Object を相互変換します。
	HTML 書式のデータと JavaScript Object を相互変換します。

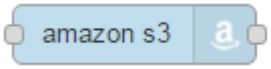
	JSON 書式のデータと JavaScript Object を相互変換します。
	XML 書式のデータと JavaScript Object を相互変換します。
	入力データが変化した場合のみ出力します。

3-5. social ノード

	電子メールを待ち受け、入力データもしくはイベントとします。
	Twitter からのメッセージを待ち受け、入力データもしくはイベントとします。
	入力データを設定された電子メールアドレスに送付します。
	入力データを Twitter へ出力します。
	Google plus に対する入出力を提供します。
	Google places に対する入出力を提供します。
	Google calendar に登録されている次のイベントを返します。

3-6. storage ノード

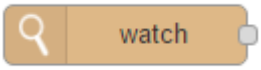
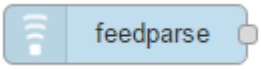
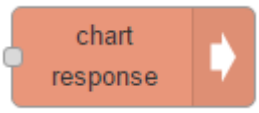
	設定されたファイルの末尾を入力データとします。
	入力データに示されるファイルを開き、その内容を出力します。
	Amazon S3 からデータを取り込みます。
	設定されたファイルにデータを出力します。

	Amazon S3 に対する入出力を提供します。
	Amazon S3 にデータを出力します。

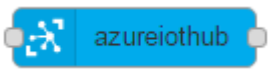
3-7. analysis ノード

	入力データを AFINN-111 単語リストを用いて感情分析（肯定的／否定的／中立）を行い出力します。

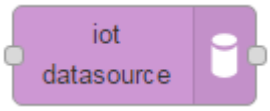
3-8. advanced ノード

	ディレクトリまたはファイルの更新を監視し、入力イベントとします。
	RSS/Atom を監視し、Web コンテンツの更新を入力イベントとします。
	システムのコマンドを実行し、その出力を返します。
	Google chart にグラフ描画をリクエストします。
	Google chart のグラフ描画を出力します。

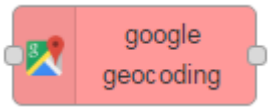
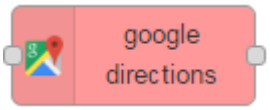
3-9. cloud ノード

	入力データを Azure IoT Hub へ出力します。
	入力データを Azure IoT Hub へ出力します。
	入力データに示されるデバイスを Azure IoT Hub へ登録します。

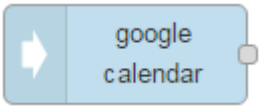
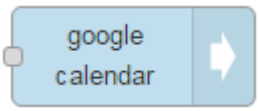
3-10. GatewayKit ノード

	入力データをダッシュボードアプリケーションへ出力します。

3-11. location ノード

	Google geocording を用いて入力データをジオコーディング（住所情報と地理的座標の相互変換）し、出力します。
	Google directions を用いて入力された出発地点と目的地点の経路を出力します。

3-12. Google ノード

	Google calendar のイベント（予定の通知）の発生を待ち受けます。
	Google calendar に新しい予定（イベント）を登録します。

第 4 章　ノード操作サンプル

本項では、BLE センサー（ALPS 電気社製　IoT Smart Module）のデータを受け Node-RED で処理します。

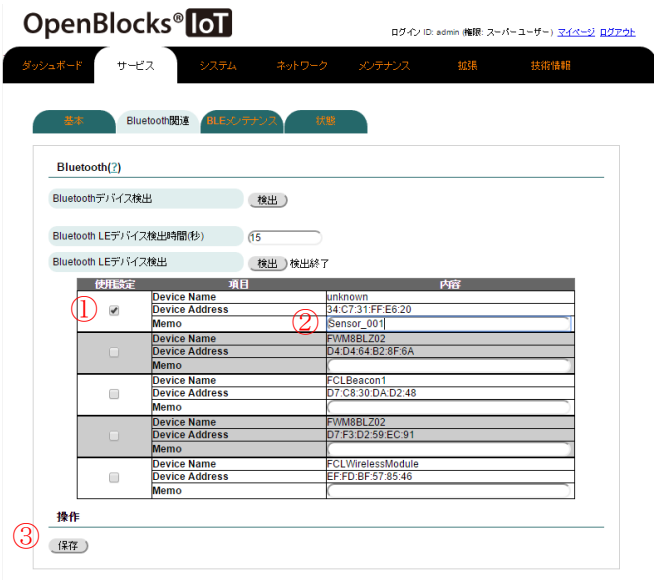
4-1. 事前準備

BLE センサーのデータを受け Node-RED へ送る設定を行います。

1. WEB UI の「サービス」→「Bluetooth 関連」タブにて Bluetooth LE デバイスの検出を行います。

Bluetooth LE センサーデバイの電源を入れ、「Bluetooth LE デバイス検出」の検出ボタン①を押してください。

2. 検出された Bluetooth LE デバイスの一覧から使用するデバイスを選択します。



「使用設定」のチェックボックス①をチェックし、必要に応じ「Memo」②を記載、保存ボタン③をクリックします。

3. 選択したデバイスが、一覧に表示されていることを確認します。

一覧に選択したデバイスが表示されていること①を確認します。

4. WEB UI の「サービス」→「基本設定」タブにて「データ収集」と「PD Handler BLE」を使用する設定を行います。

「データ収集」①と「PD Handler BLE」②を”使用する”に設定し、保存ボタン③を押してください。

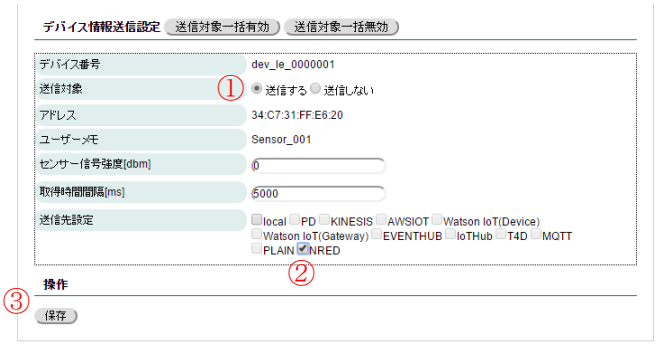
5. WEB UI の「サービス」→「収集設定」タブの「送信先設定」にて Node-RED へ送る設定を行います。

「node red（NRED)」①を”使用する”に設定します。

他のパラメータはデフォルト値のままとしてください。

保存ボタンは次のステップでクリックします。

6. WEB UI の「サービス」→「収集設定」タブの「デバイス情報送信設定」にて「デバイス番号」dev_le_0000001 のデータを Node-RED へ送る設定を行います。



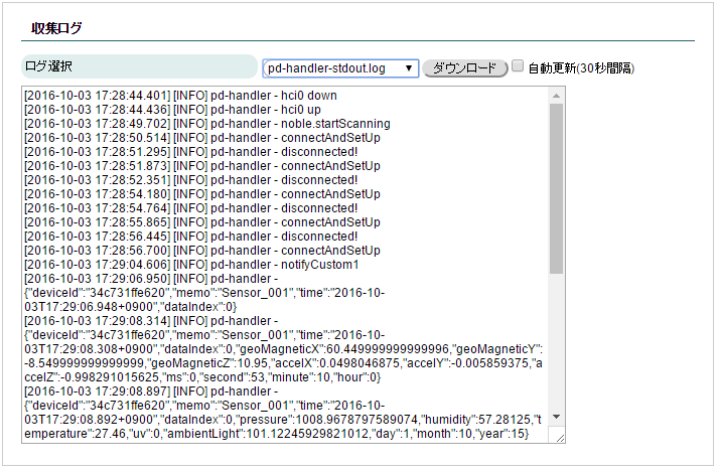
「送信対象」①を”送信する”に設定します。

「送信先設定」の”NRED”チェックボック②をチェックします。

他のパラメータはデフォルト値のままとしてください。

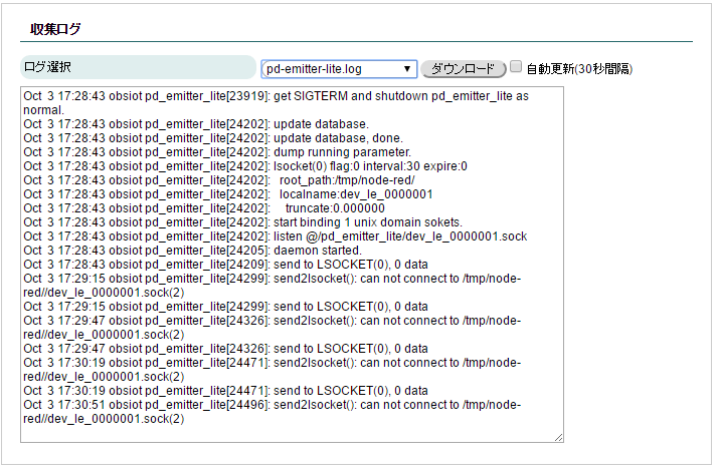
保存ボタン③をクリックします。

7. WEB UI の「サービス」→「収集ログ」タブで動作を確認します。



「ログ選択」にて「pd-handler-stdout.log」を選択します。

「{“deviceId”: 」で始まる JSON 文字列があることを確認します。



「ログ選択」にて「pd-emitter-lite.log」を選択します。

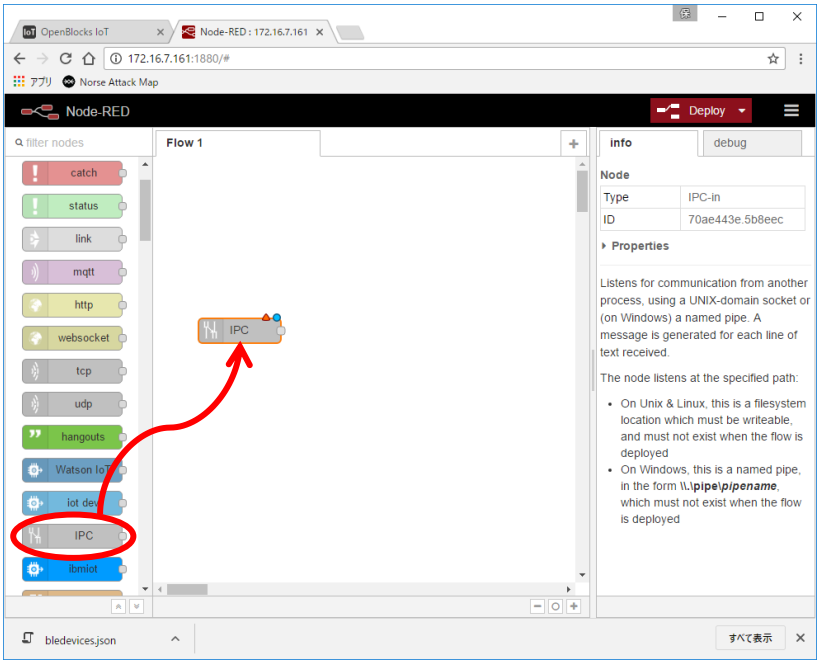
この時点では、Node-RED 側の UNIX ドメインソケットが用意されていないため、接続エラーが発生しています。

4-2. 単純なデバッグ

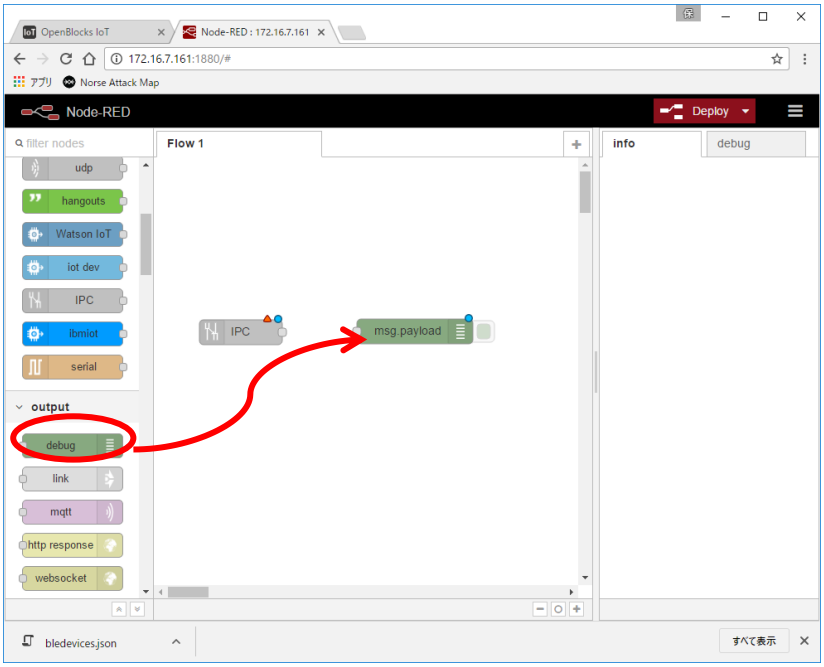
1. Node-RED にアクセスします。（第３章参照）

   http://192.168.254.254:1880/

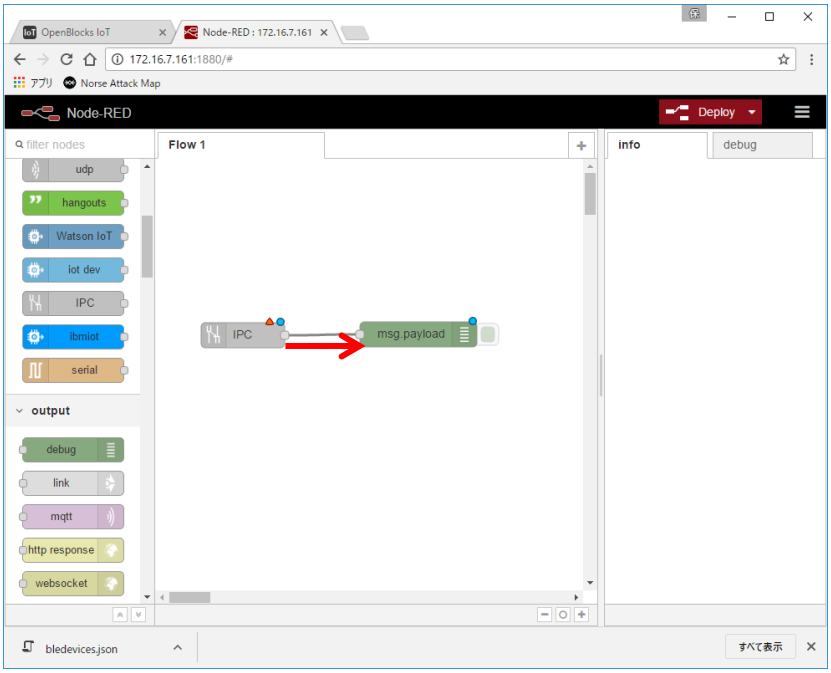
2. Input ノードパレットから IPC-in node をドラッグしてシートにドロップします。



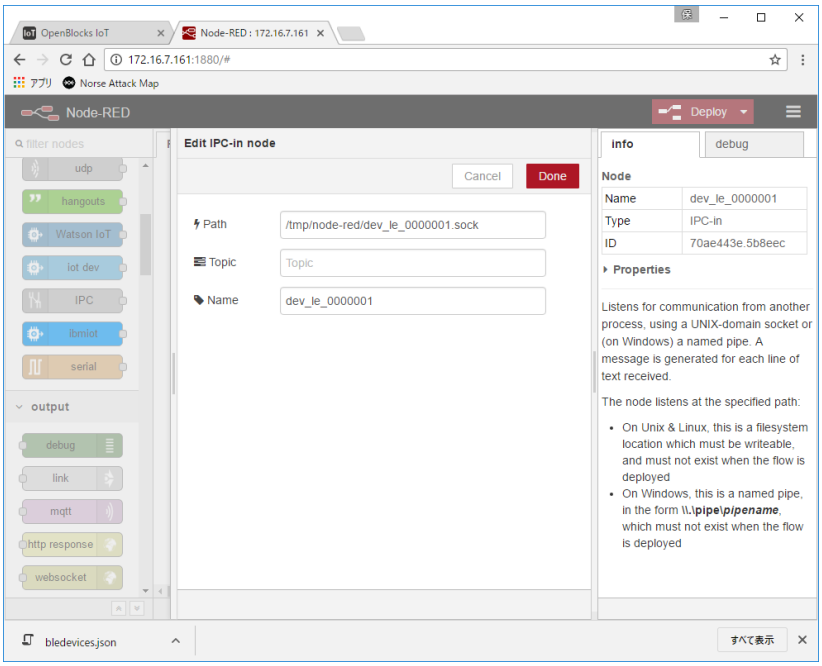
3. Output ノードパレットから debug node をドラッグしてシートにドロップします。



4. IPC-in node と debug node を結線します。



5. IPC-in node をダブルクリックし、Path と Name を設定し、Done ボタンをクリックします。



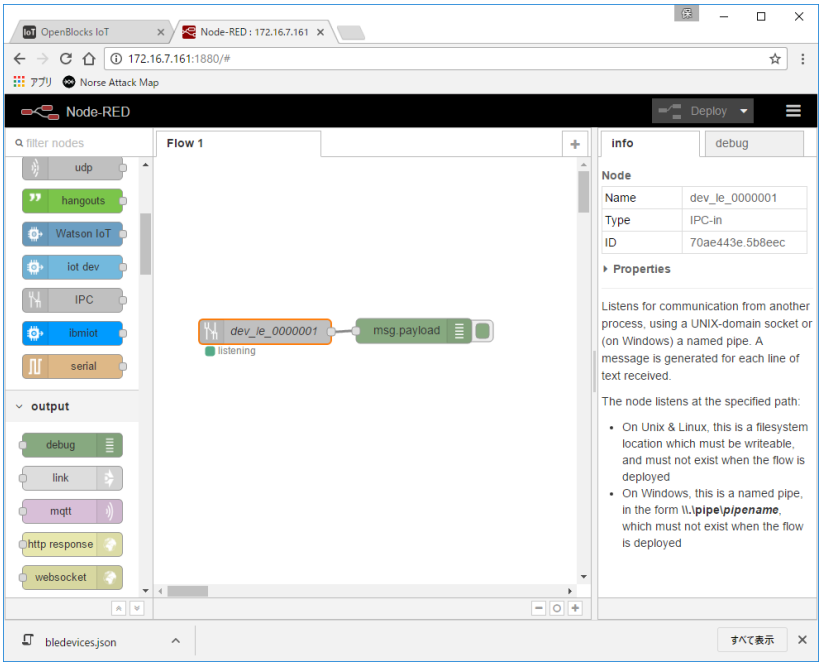
ここで、Path は、4-1 事前設定の 1 に図示される「ソケットパスプレフィックス」と同 2 に図示される「デバイス番号」に .sock を付加したものとします。

/tmp/node-red/dev_le_0000001.sock

Name は、何でも構いませんが本例では dev_le_0000001 とします。

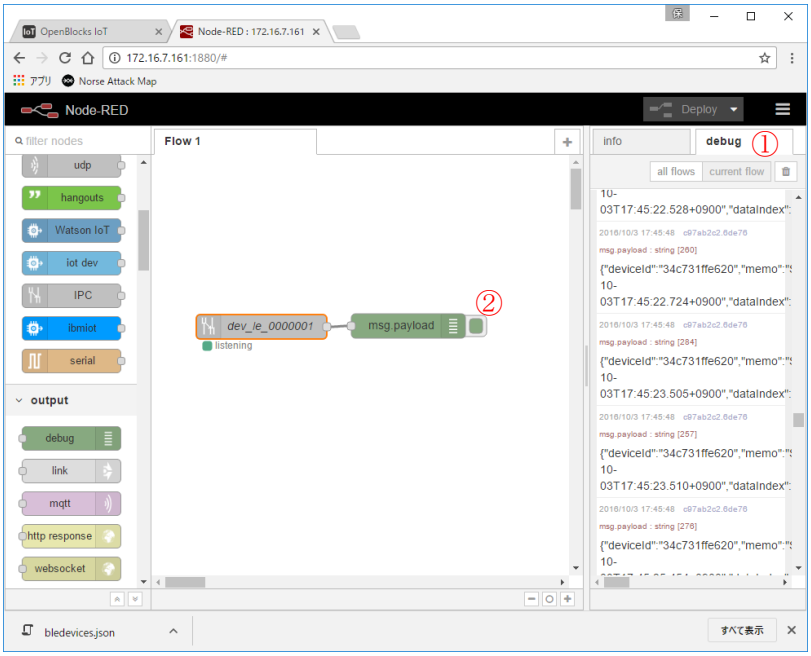
Topic は、空欄のままとします。

6. Deploy ボタンをクリックします。



Deploy が完了すると、Deploy ボタンの背景が赤から黒に変わります。

7. ノード表示をデバック表示に切り替え（①をクリック）、debug node を active にします（②をクリック）。



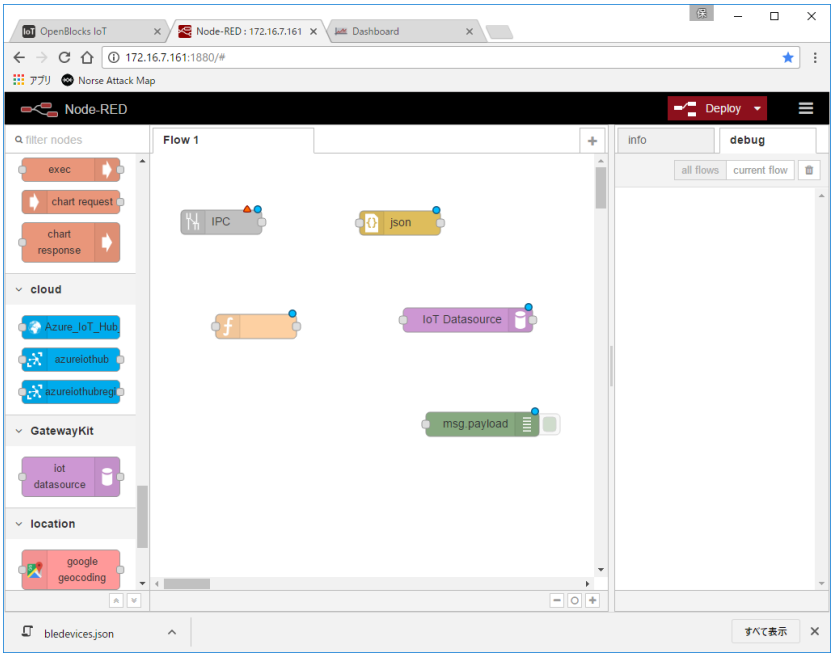
デバック表示にビーコンから得られたデータが表示されます。

4-3. 温度データをグラフに表示する

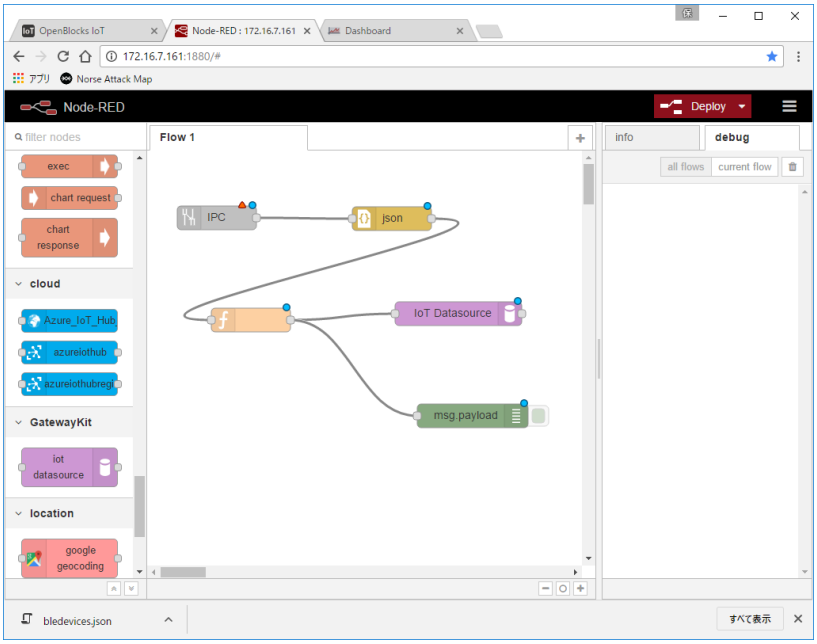
IoT datasource ノードを用いて、BLE センサーから取り込まれる温度データをグラフ表示します。

1. Node-RED にアクセスします。（第 3 章参照）
   http://192.168.254.254:1880/

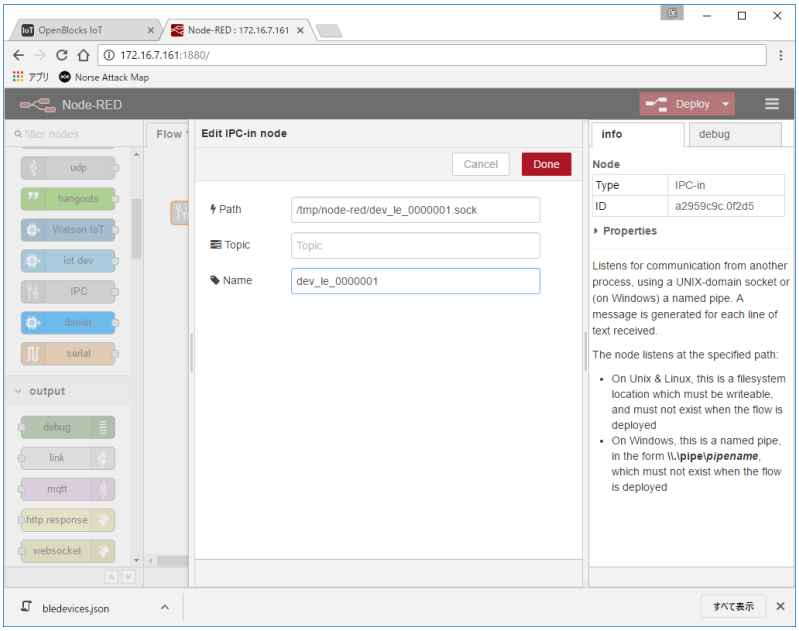
2. 下図に示す通り IPC-in node と json node, function node, iot datasource node, debug node をそれぞれドラッグしシートにドロップします。



3. 下図に示す通り IPC-in node と json node, function node, iot datasource node, debug node の間を結線します。



4. IPC-in node をダブルクリックし、Path と Name を設定し、Done ボタンをクリックします。



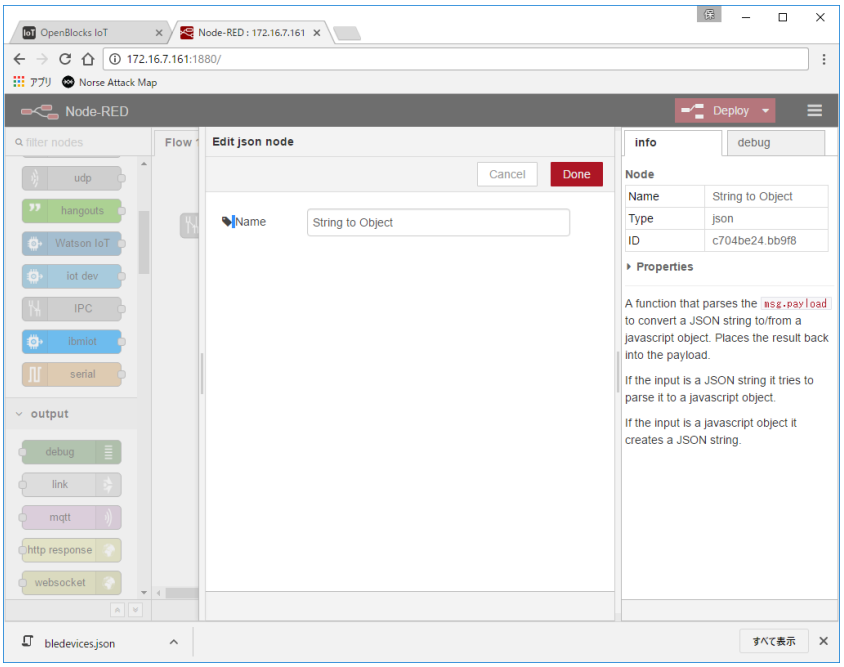
ここで、Path は、4-1 事前設定の 1 に図示される「ソケットパスプレフィックス」と同 2 に図示される「デバイス番号」に .sock を付加したものとします。

/tmp/node-red/dev_le_0000001.sock

Name は、何でも構いませんが本例では dev_le_0000001 とします。

Topic は、空欄のままとします。

5. JSON node をダブルクリックし、適当な Name を設定し、Done ボタンをクリックします。

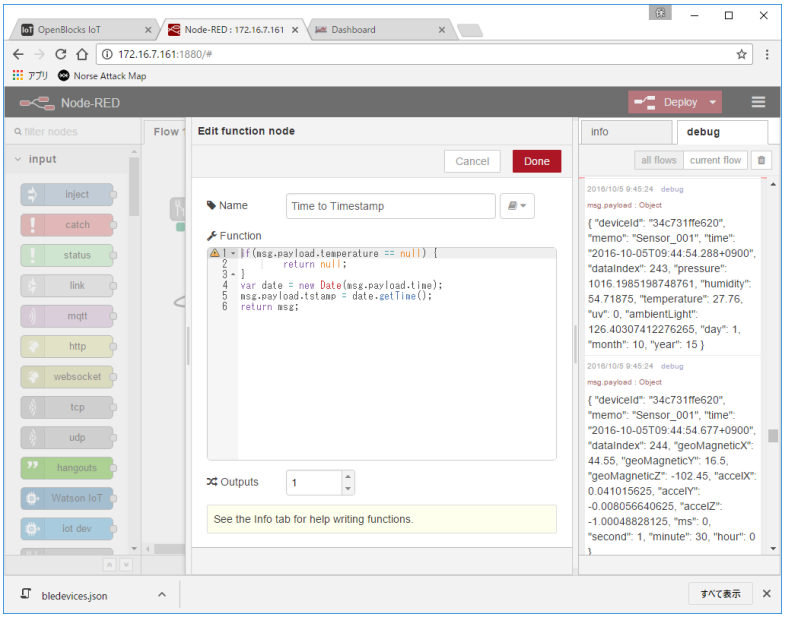


特に指定が無い限り OpenBlocks IoT ファミリーに標準搭載されているデータ取り込みアプリケーション（PD-Handler）の出力フォーマットは JSON 形式です。

従って IPC-in node を用いて PD-Handler のデータを取り込み Node-RED でデータを処理する場合、JSON node は必須となります。

6. function node をダブルクリックし、適当な Name を設定、Function として次の JavaScrip を記述し、Done ボタンをクリックします。

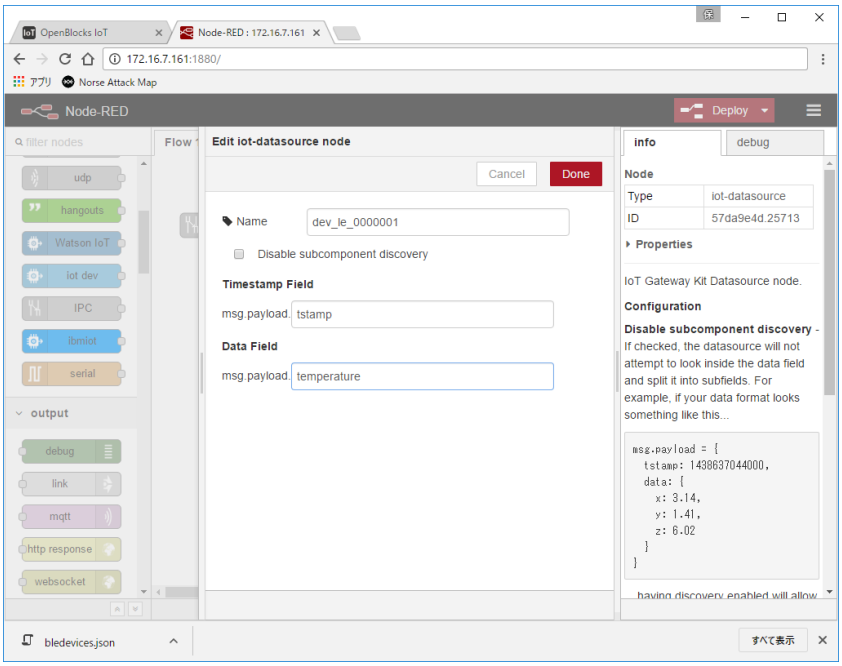
```
/* ① */
if(msg.payload.temperature == null) {
        return null;
}

/* ② */
var date = new Date(msg.payload.time);
msg.payload.tstamp = date.getTime();

return msg;
```



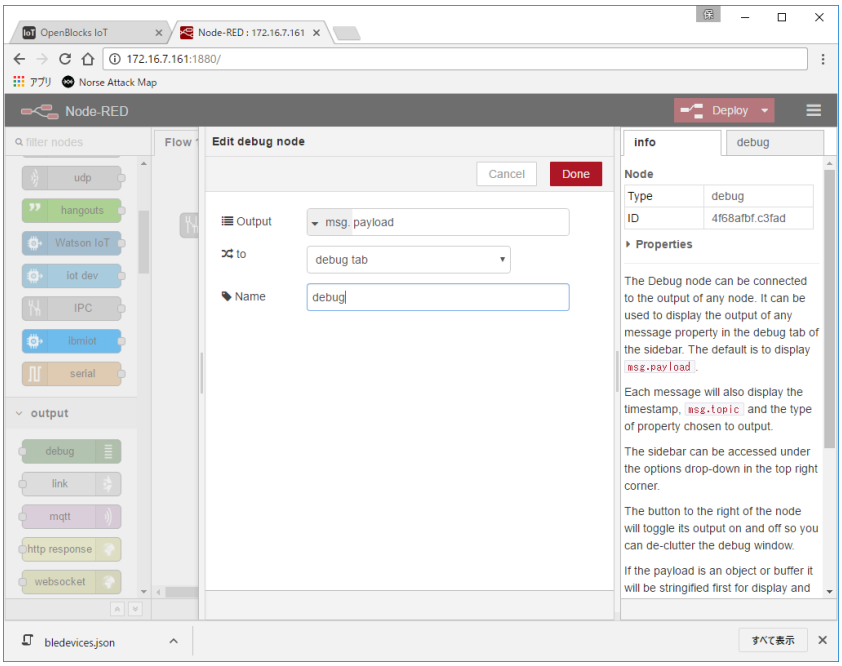
① ALPS 電気社製 IoT Smart Module は、温湿度等の環境データと加速度等のモーションデータを別々のタイミングで送信するため、温度情報を含まないデータ(msg)をブロックします。

② データに含まれる RFC3339 形式のタイムスタンプ(msg.payload.time)から、iot Datasource node で使用する UNIX 形式のタイムスタンプに変換します。

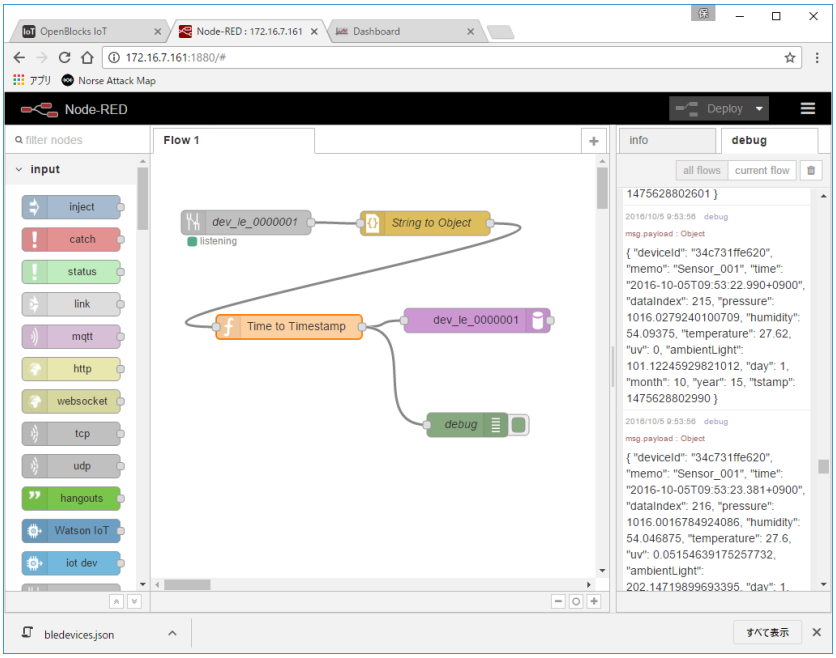
7. Iot Dataresource node をダブルクリックし、Name として dev_le_0000001 を設定、Timestamp Field として msg.payload.tstamp 、Data Field として msg.payload.temperature を設定し、Done ボタンをクリックします。



8. Debug node をダブルクリックし、適当な Name を設定し、Done ボタンをクリックします。

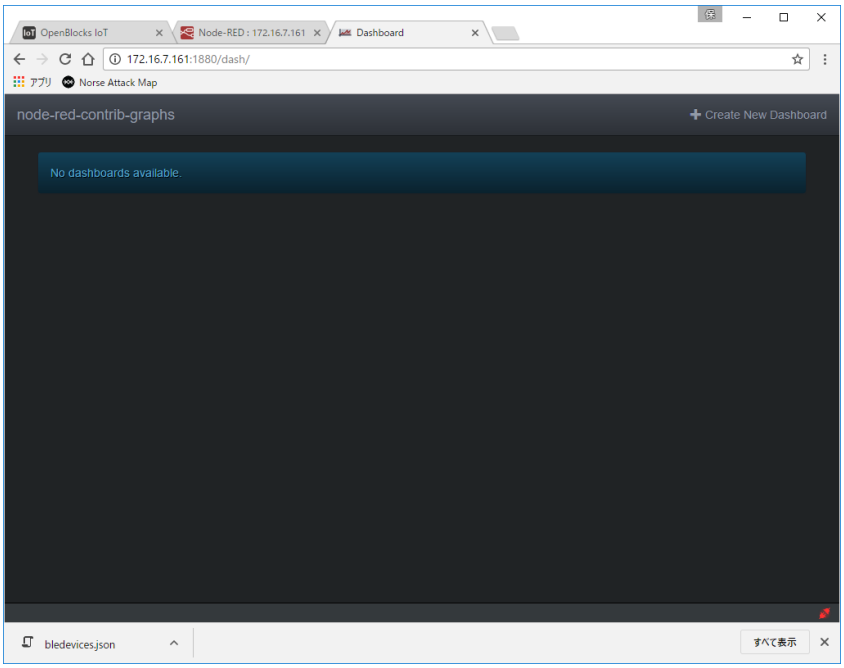


9. Deploy ボタンをクリックします。

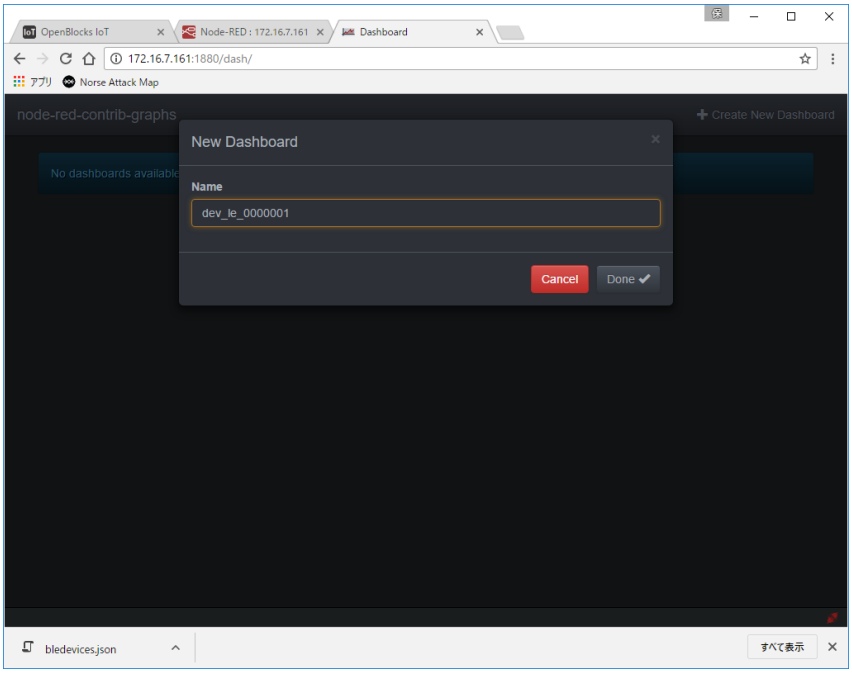


10. 新しいブラウザタブを開き Iot Datasource node のダッシュボードにアクセスします。

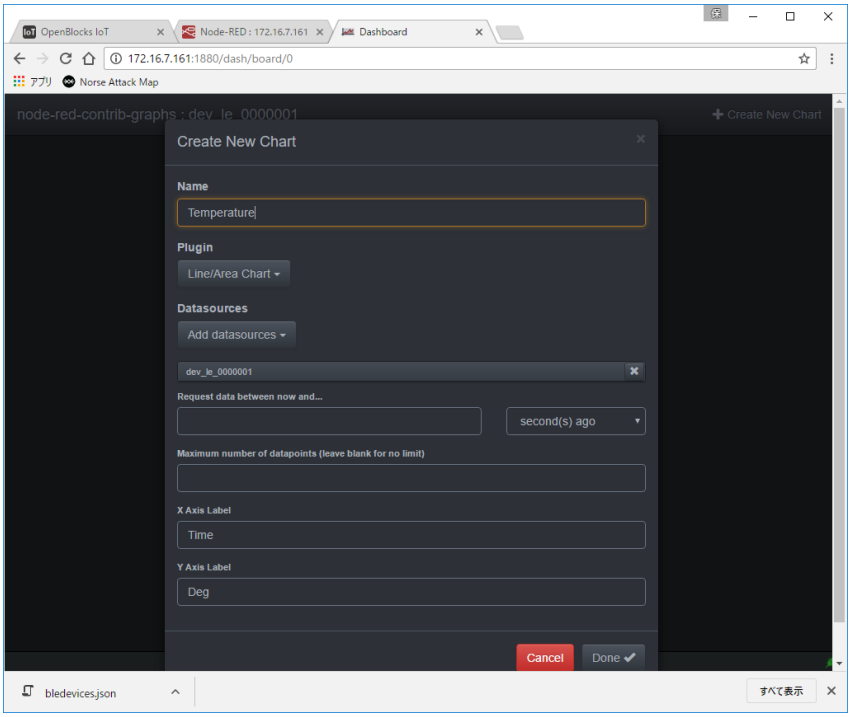
http://192.168.254.254:1880/dash/



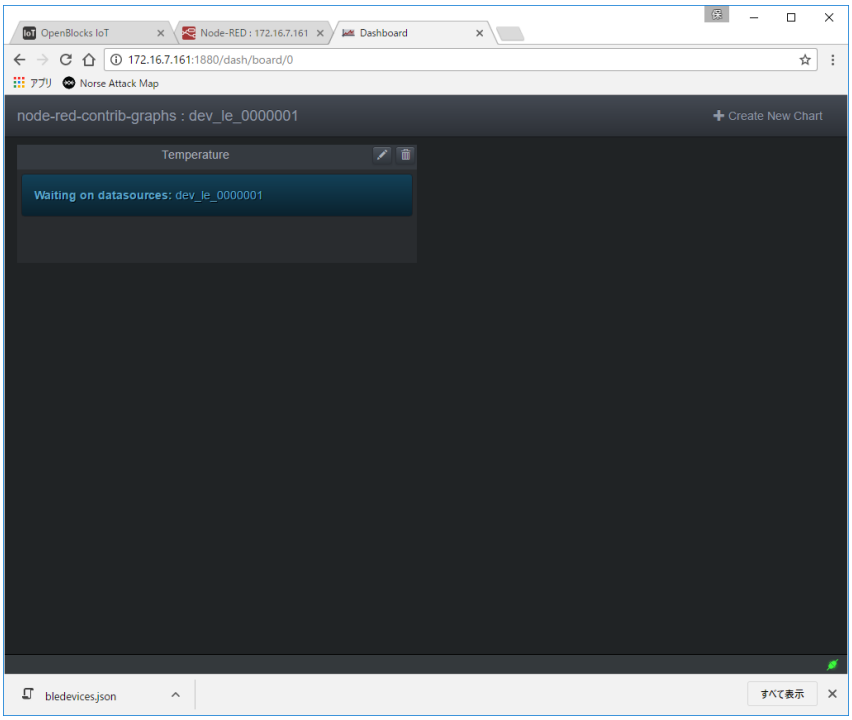
11. +Create New Dashboard をクリックし、Name として dev_le_0000001 を設定し、Done をクリックします。



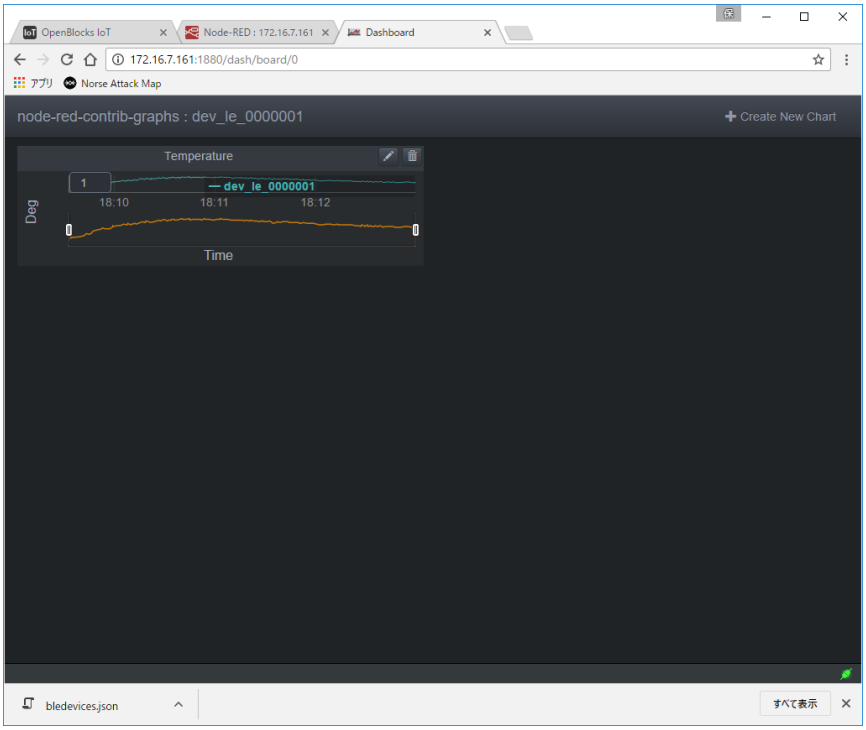
12. +Create New Chart をクリックし、適当な Name を設定し、Plugin として Line/Area Char を選択、Add datasource として dev_le_0000001 を選択、X Axis Label として”Time”、Y Axis Label として”Deg”を設定し、Done をクリックします。



13. データ待ち状態の画面が表示されます。



14. データが取り込まれるとグラフが表示されます。



OpenBlocks IoT VX1 向け Node-RED スターターガイド
（2016/10/17 第 1 版）

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