さくらの IoT Platform を使ってみよう

DAY
2016/11/27

COMPANY
さくらインターネット株式会社

DEPARTMENT
コミュニティマネージャー

NAME
法林 浩之

Facebook　法林 浩之

Twitter　@hourin

どんな人？

- 日本UNIXユーザ会 幹事
- フリーランスエンジニア
- さくらインターネット コミュニティマネージャー
- くわしくは「法林浩之」で検索

さくらでやっていること

- 当社主催イベントの運営
- 社外イベント対応(協賛/出展/登壇/取材など)
- 10月末～12月上旬にかけて20連戦敢行中

- さくらのIoT Platformの概要
  - 開発経緯
  - 主な機能/システム構成/パートナー連携
  - 事例
  - β版の販売について
- 実際に使ってみる
  - マイコンおよびプログラムの構築
  - さくらのIoT Platformの設定
  - Webサービスとの連携

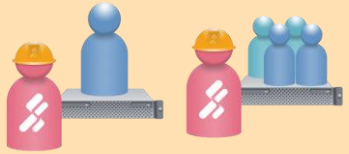
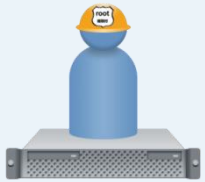
さくらインターネットについて

大阪本社

東京支社

商号	さくらインターネット株式会社 （SAKURA Internet Inc.）
代表取締役	田中 邦裕
設立	1999年8月17日 （サービス開始: 1996年12月23日）
資本金	8億9,530万円
事業内容	インターネットでのサーバの設置およびその管理業務 電気通信事業法に基づく電気通信事業 マルチメディアの企画ならびに製作・販売 インターネットに関するコンサルティング
従業員数	339名 （2016年3月末）
所属団体	特定非営利活動法人日本データセンター協会 （JDCC） 社団法人コンピュータソフトウェア協会(CSAJ) 社団法人日本ネットワークインフォメーションセンター （JPNIC） 社団法人インターネットプロバイダー協会 （JAIPA） グリーン・グリッド （The Green Grid） IPv6普及・高度化推進協議会 社団法人電子情報技術産業協会 グリーンIT推進協議会 ASP・SaaSインダストリ・コンソーシアム ホスティングビジネス研究会

データセンターにまつわるサービスのすべてを提供

レンタルサーバ	仮想サーバ	専用サーバ	ハウジング
			
さくらのレンタルサーバ **さくらのマネージドサーバ**	**さくらのVPS** **さくらのクラウド**	**さくらの専用サーバ**	**ハウジング** **リモートハウジング**
1台のサーバを複数の契約者で共有して利用するサービス	1台のサーバを仮想的に分割し、分割された領域を占有できるサービス	顧客が物理サーバ1台を丸ごと占有するサービス	顧客が所有する機器類を設置するスペースと回線、電源などを貸与するサービス

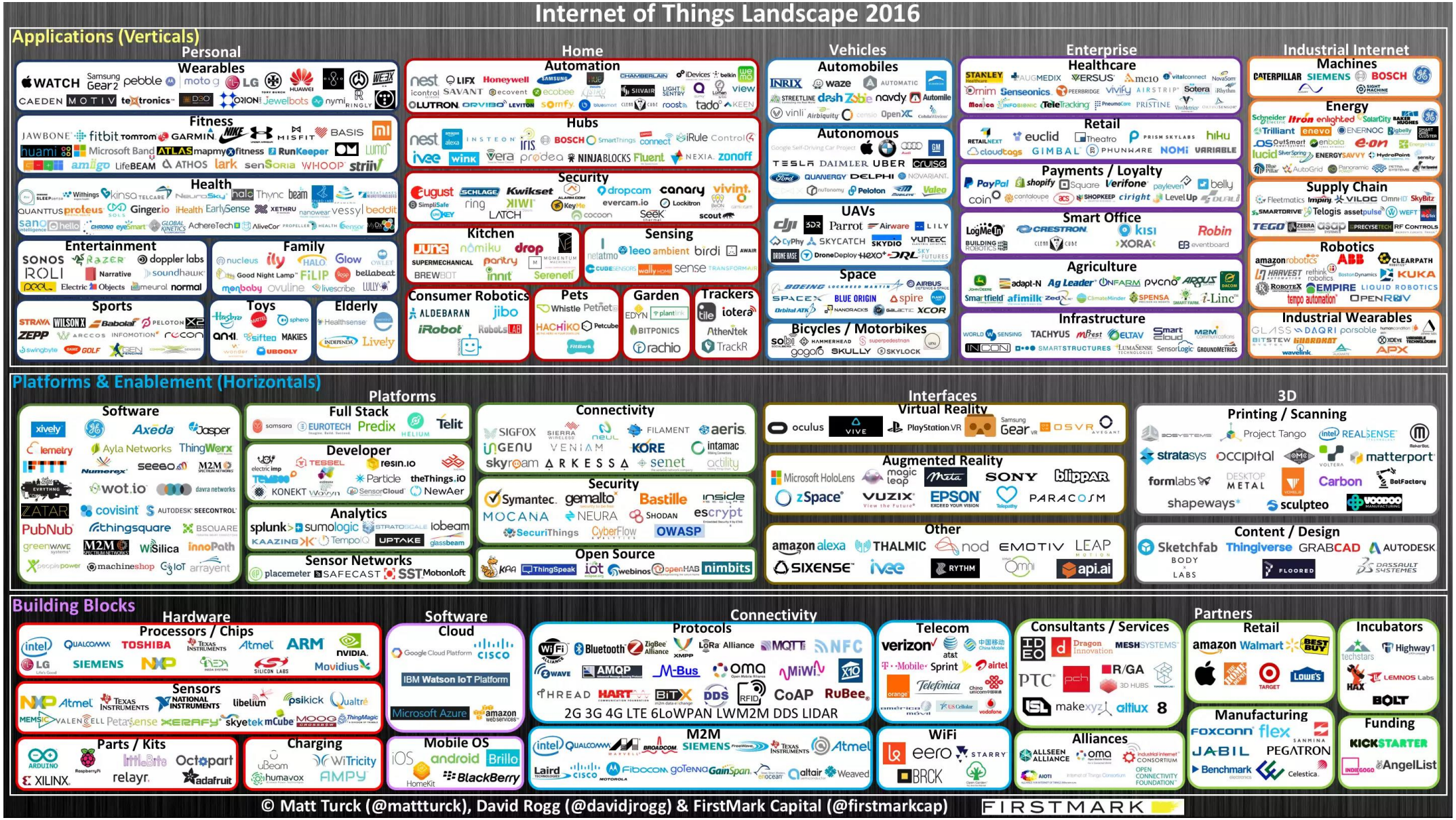
ホスティング ← → コロケーション

さくらのIoT Platform β NEW

さくらのサービス上で稼働

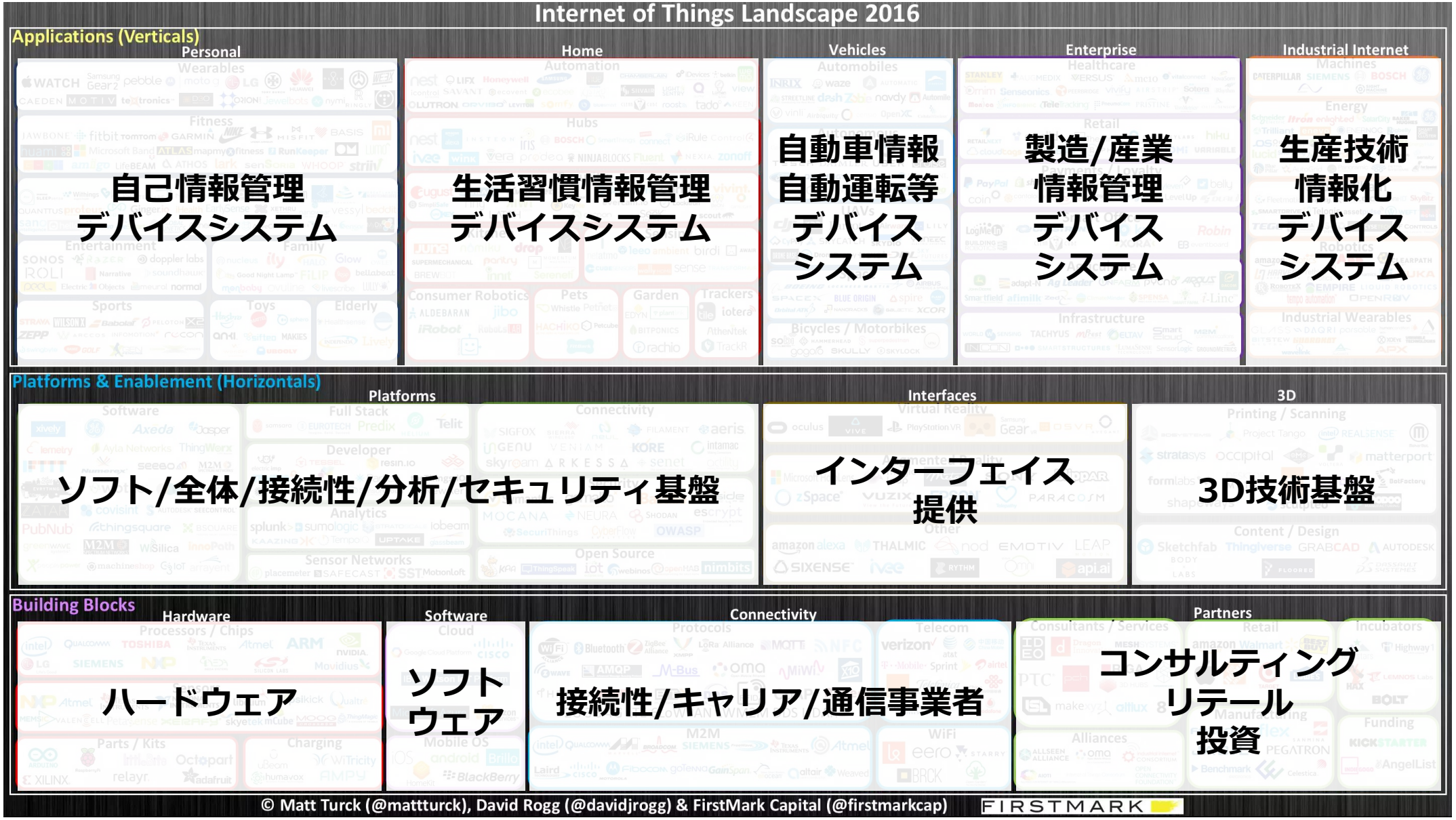
"IoT"とは何か？

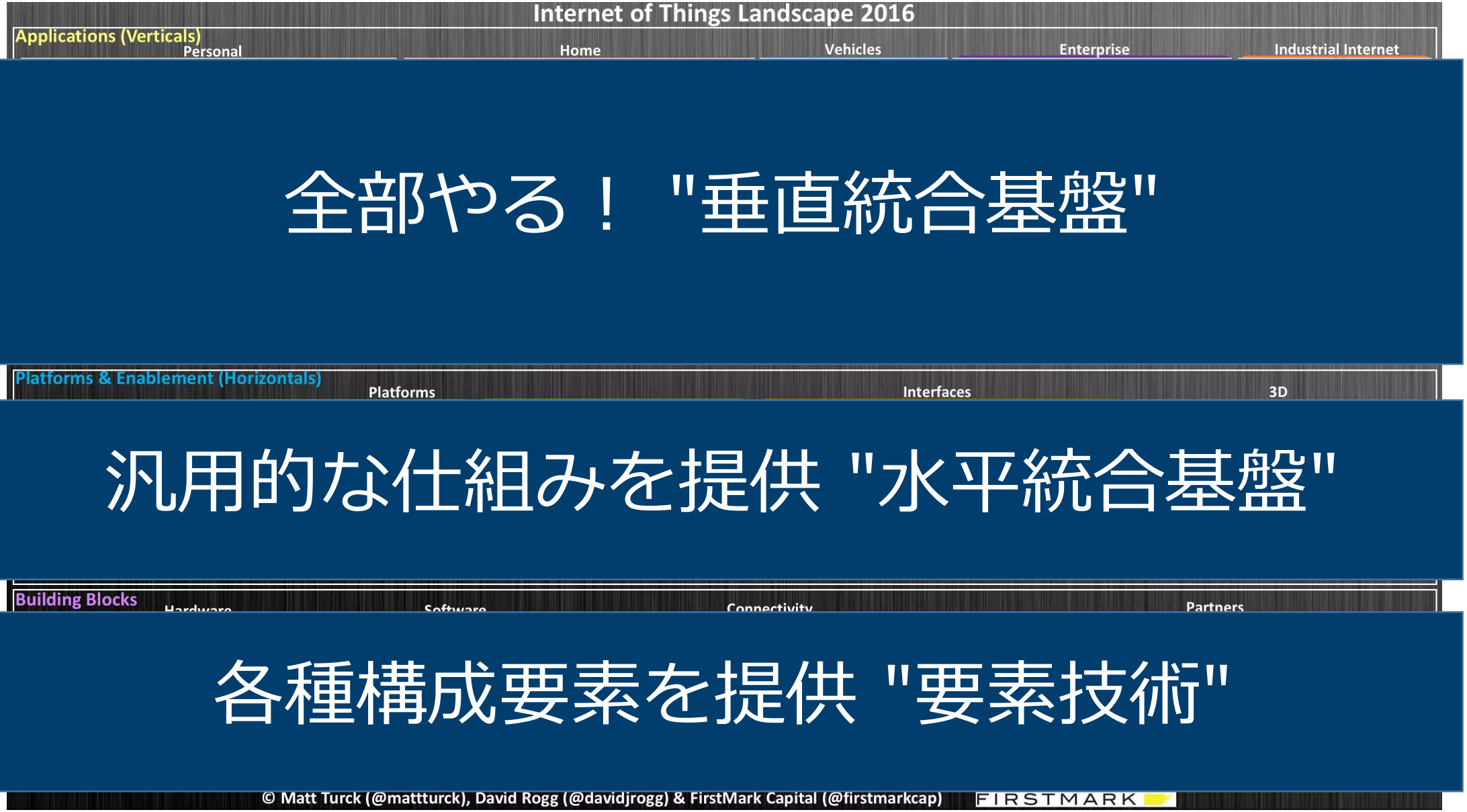
IoTと呼ばれているもの



"インターネット"と同じで広すぎる範囲

IoTと呼ばれているもの

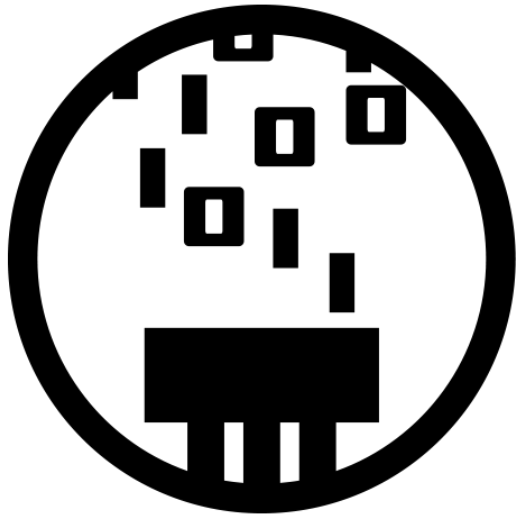




全部やる！ "垂直統合基盤"

汎用的な仕組みを提供 "水平統合基盤"

各種構成要素を提供 "要素技術"



さくらのIoT Platform β

通信モジュール
LTE閉域網
データ保存/連携サービス

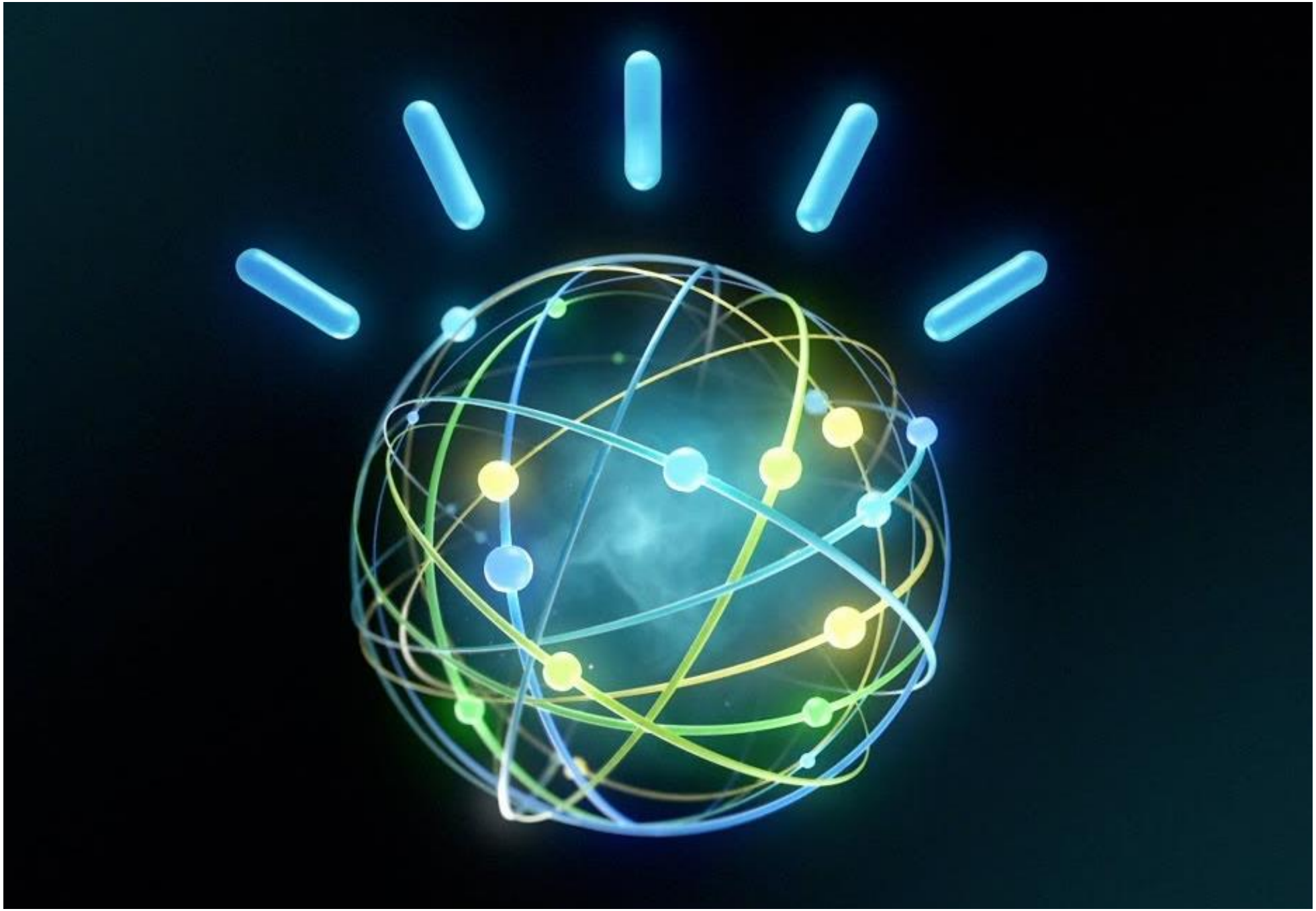
統合型プラットフォーム

他のIoTプラットフォームとの違い

AlphaGo

Ponanza

※さくらのサーバで機械学習しています
http://ascii.jp/elem/000/001/171/1171630/



特化型AI（囲碁/将棋）
コグニティブコンピューティング

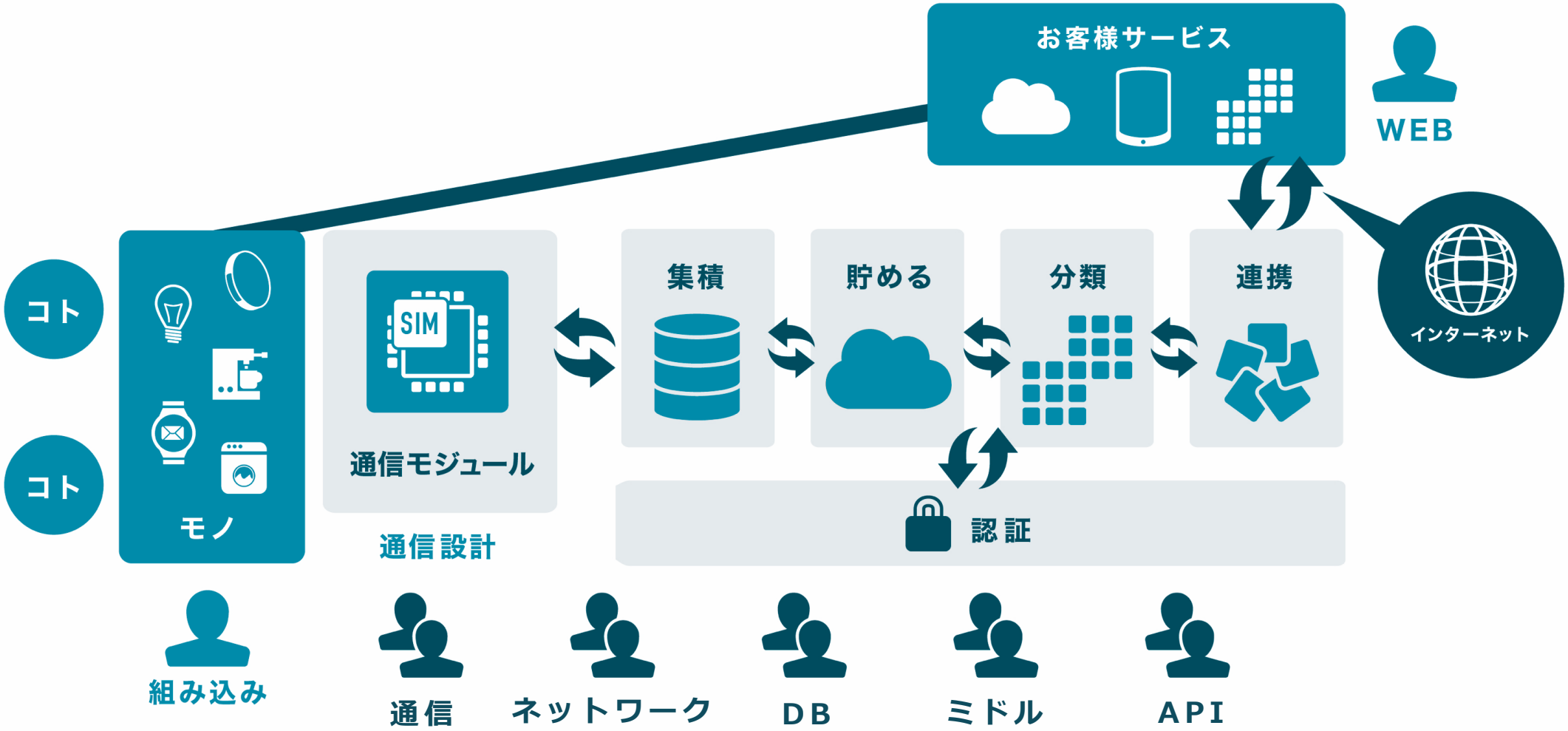
データさえあれば
何でも分析できます！！

どうやって
データ
集めるの？

そこに、さくら

さくらのIoT Platformβ
概要

すべてを自社でやるのは現実的でしょうか？

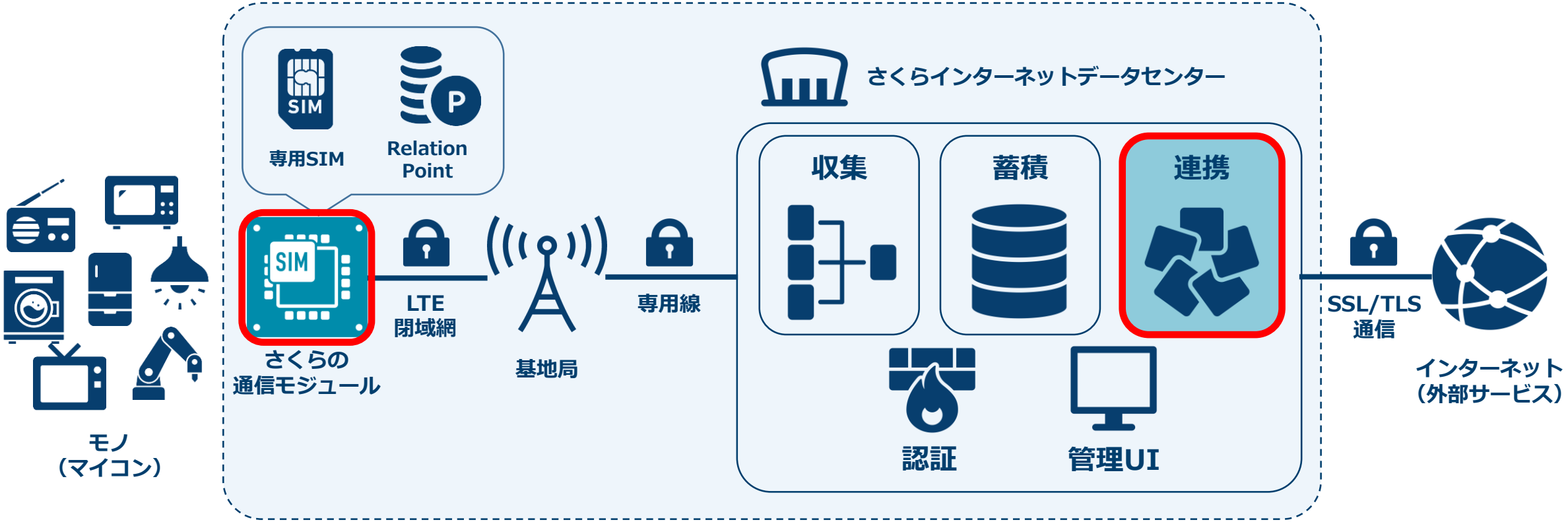


ネットワークとデータをやり取りしたいだけなのに…
やらなければならないことが多すぎる

すべてを自社でやるのは現実的でしょうか？

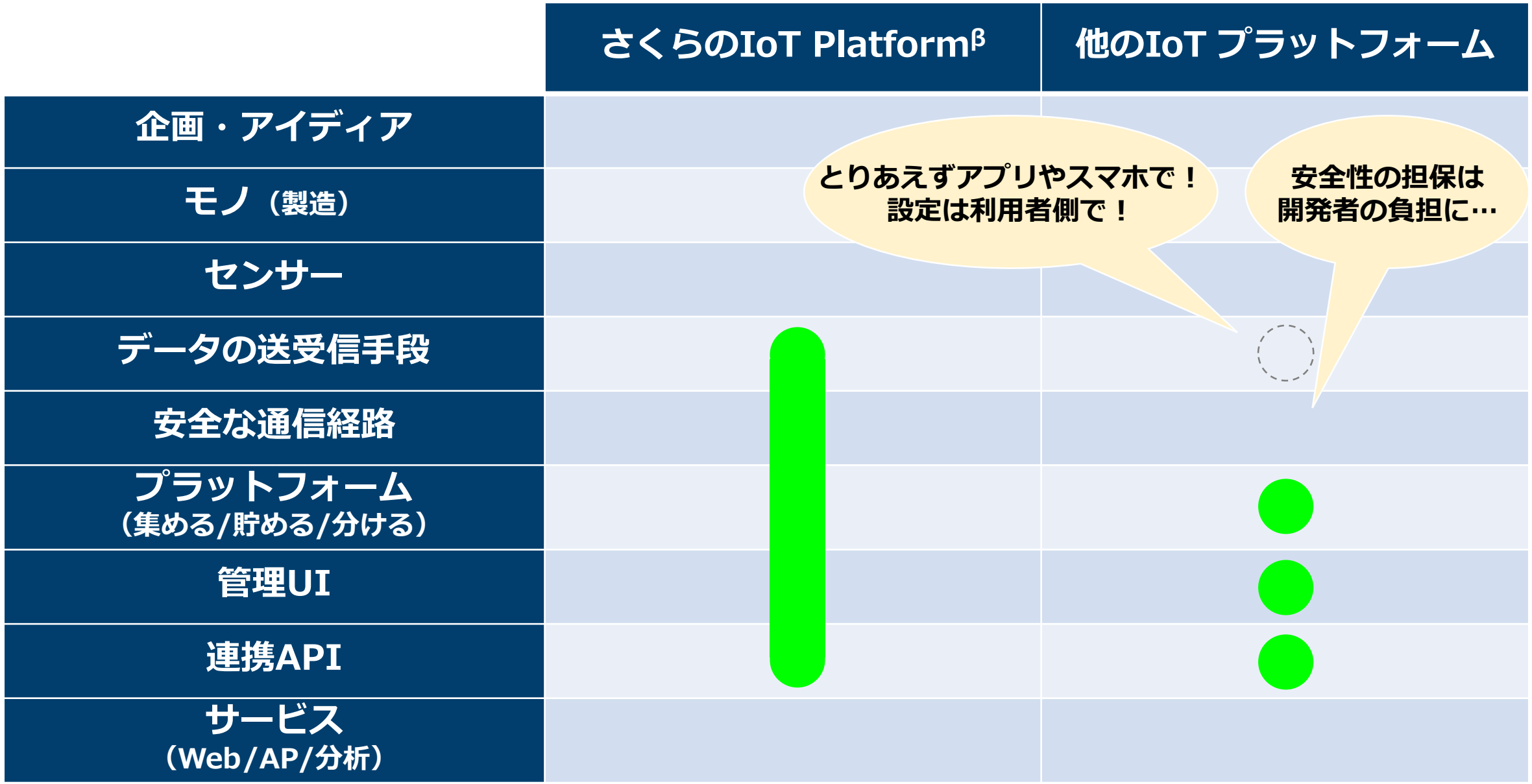
既存の事業領域/スキルセットの大幅な変更なく
モノ/サービスづくり、連携に注力可能

さくらのIoT Platformβ ご提供範囲



モノに組み込む「さくらの通信モジュール」から
データを取り出す「連携サービス」までを統合して提供

他のIoTプラットフォームとの差異



他のIoTプラットフォームとの差異

	さくらのIoT Platformβ	他のIoT プラットフォーム
企画・アイディア		
モノ（製造）		
センサー		
データの送受信手段		
安全な通信経路		
プラットフォーム（集める/貯める/分ける）		
管理UI		
連携API		
サービス（Web/AP/分析）		

「データを迎えに行く」という発想

- モノからのアウトプットだけでなくモノへのインプットも可能
- モノに組み込めば、電源を入れるだけで利用可能
- 利用者に接続の知識や現地の有線/無線LAN環境も不要

さくらのIoT Platformβ
機能詳細

1. IoTデバイス、サービスの“**開発工数削減**”が可能
2. “**SDカードほぼ2枚分**”に収まるコンパクトサイズ
3. 閉域網を利用した”**Secure＆Safety**”なネットワーク設計
4. プラットフォームサービスだから“**設計や運用は考慮不要**”
5. “**時刻提供機能**”でマイコンに現在時刻を提供
6. “**ファイル配信機能**”でマイコン側のアップデートも実現

IoTデバイス、サービスの“開発工数削減”が可能

要開発項目

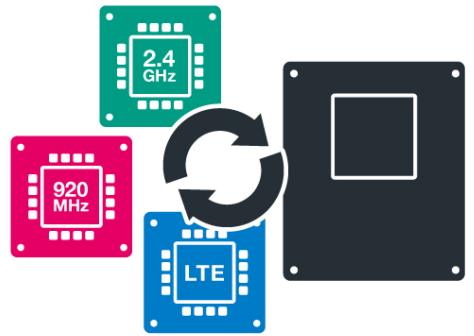
- アプリケーション
- コマンドI/F実装
- 上位プロトコル実装
- TCP/IPスタック
- モデムコマンド制御
- UART制御
- 省電力制御

削減工数

従来の通信手法
I²C/SPI

さくらの
通信モジュール

アンテナ

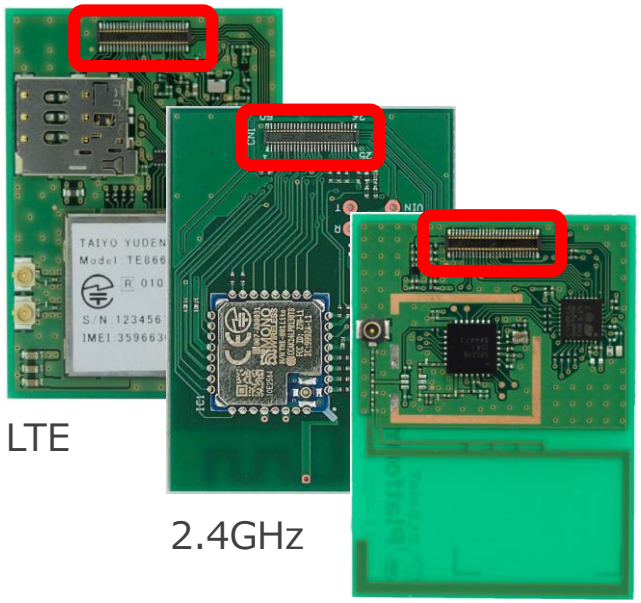


複数の無線規格に対応

「作らなければならないもの」より「作りたいもの」に注力が可能

IoTデバイス、サービスの“開発工数削減”が可能

量産性に配慮した
“**基板間コネクタ**”を採用



LTE

2.4GHz

920MHz(LoRa)

方式	GW	特徴	通信可能レンジ	伝送速度	消費電力
LTE	不要	単独使用可	キャリア網内どこでも	速い	大きい
2.4GHz帯	必要	短距離大容量	数百メートル（最大1Km程度）	速い	小さい
920MHz帯	必要	長距離小容量	数キロメートル（最大10km程度）	遅い	小さい

共通I/Fおよび寸法のため複数規格のモジュールの対応が容易

IoTデバイス、サービスの“開発工数削減”が可能


```
{
   "module": "XXXXXXXXX",
   "type": "channels",
   "datetime": "2016-06-01T12:21:11.628907163Z",
   "payload": {
      "channels": [{
            "channel": 1,
            "type": "i",
            "value": 1,
            “datetime": "2016-06-01T10:21:11.628907163Z",
         },
          {
            "channel": 2,
            "type": "b",
            "value": "0f1e2d3c4b5c6b7a"
            “datetime": "2016-06-01T11:21:11.628907163Z",
          }
      ]
   }
}
```

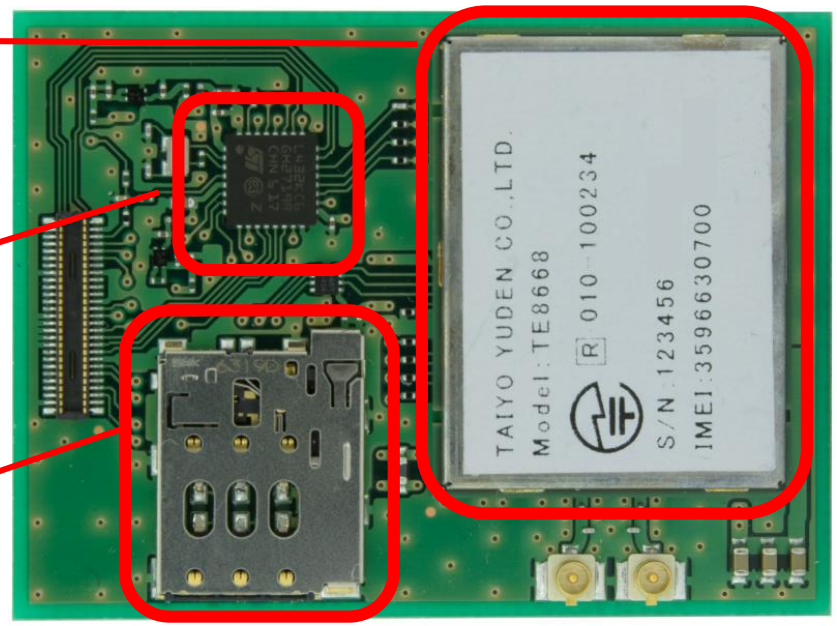

データ活用は暗号化された経路から扱いやすいJSONフォーマットで

“SDカードほぼ2枚分”に収まるコンパクトサイズ

LTEモデムチップ

通信制御用MCU

SIMカードコネクタ



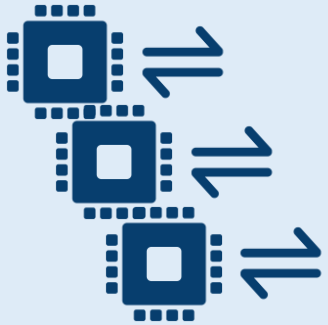
“さくらの通信モジュール”にモノ側の通信に必要なすべてを凝縮

閉域網を利用した”Secure＆Safety”なネットワーク設計

通信モジュール+SIMカードをセットで提供することで安全性を確保

プラットフォームサービスだから“設計や運用は考慮不要”

サービスの“設計”



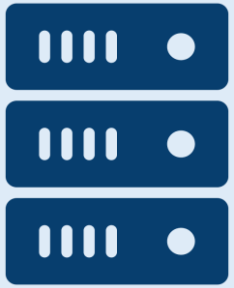
データの収集

データの蓄積

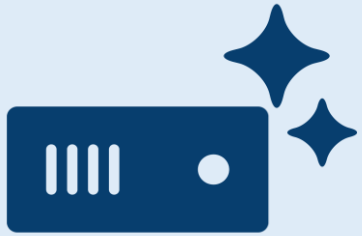
データの連携

セキュリティ

サービスの“運用”



ラージスケール対応



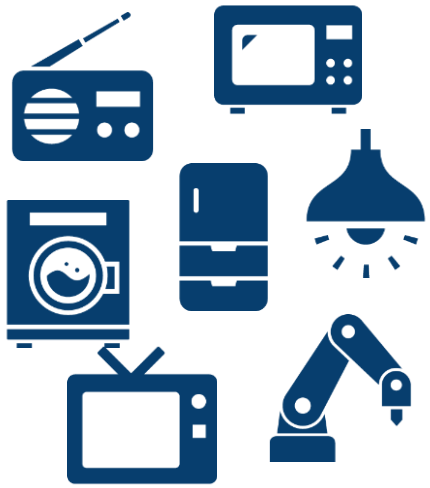
アップデート

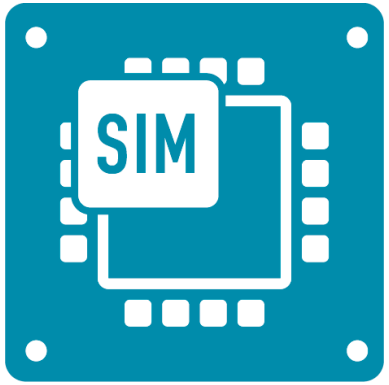
障害切り分け、復旧

IoTサービスに不可欠な”設計”や”運用”はさくらインターネットが対応

“時刻提供機能”でマイコンに現在時刻を提供

マイコンへの適用





時刻情報の要求

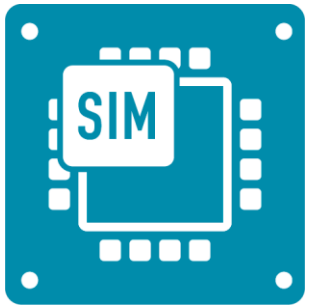
正確な時刻の提供

マイコン側でのログ記録等が求められるデバイスやサービスに

※マイコン側で時刻情報要求の制御が必要です

“ファイル配信機能”でマイコン側のアップデートも実現

マイコンへの適用



さくらの
通信モジュール

モジュールへの配信

さくらの
IoT Platform

ファイルのアップロード

管理者

ソフトウェア的な問題はアップデートで対応可能

※マイコン側でのファームウェア書換制御が必要です

選べる複数のデータ保存方式

オープン

オープンデータとして活用可能にいただくことで
データ保存のコストをゼロにできます。

クローズ

公開したくないデータはユーザのみ閲覧権限を
付与いただくことができます。

プライベート

コンプライアンス対策としてデータを専用領域に
保存いただくことができます。

データ保護ポリシーに応じて適切な保存先を選択可能

リアルタイム連携

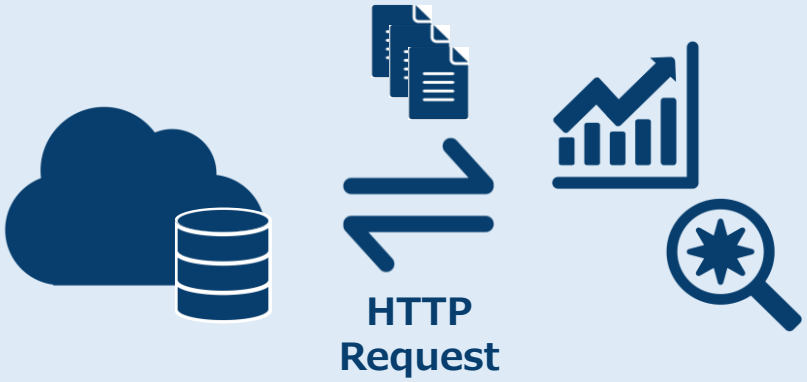
<即時性が求められるサービス向け>

通信モジュールからの受信データを即連携先に送信します。
連携先からの送信データも即通信モジュールに送信します。
提供は以下を含め、随時他社サービスも追加されます。
Outgoing Webhook/Incoming Webhook/WebSocket/MQTT(予定)

保存データの一括取得

<分析や集計等のバッチ処理向け>

通信モジュールから受信したデータを要求された時に
要求された期間分まとめて送信します。
提供はHTTP REST APIにより行われます。



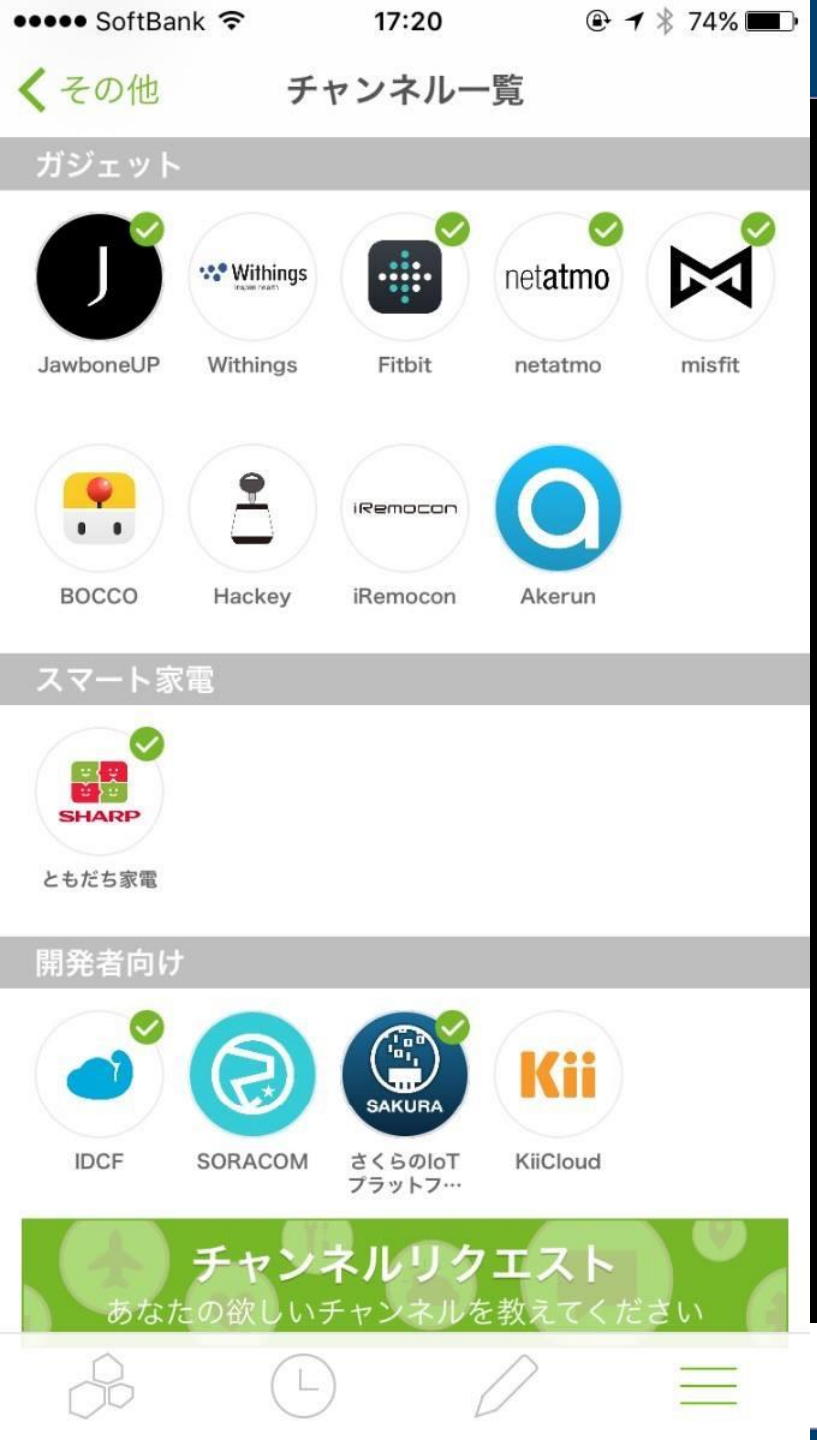
利用用途に応じて適切な連携方式を選択可能

共創パートナー

他社クラウドサービスや自社環境にも用途に応じて連携可能

myThings

インターネットサービスやIoTプロダクトを利用して、
あなたにあった便利な組み合わせをつくることができるアプリ



SAKURA Internet

myThingsに
さくらのIoT Platform β の
チャンネルが作成される予定

SAKURA Internet

トリガーを設定

例：防災速報
　地震情報が出たら
　熱中症の危険度が高まったら
　津波警報・注意報が発令されたら

アクションを設定

例：さくらのIoTプラットフォームの
1チャンネルに1という値を送信

SAKURA Internet

トリガーごとに異なるデータを
送信可能

例：熱中症の危険度が高まったら
さくらのIoTプラットフォームに
2チャンネルに2という値を送信

SAKURA Internet

例：津波警報・注意報が出たら
さくらのIoTプラットフォームの
3チャンネルに3という値を送信

さくらのIoTプラットフォームを
トリガーに使うことも可能

事例

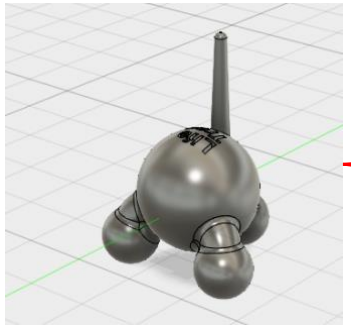
私たちは
IoTの”T”を知らなさすぎる

実証実験（PoC）を通じて連携機能/GWタイプを開発

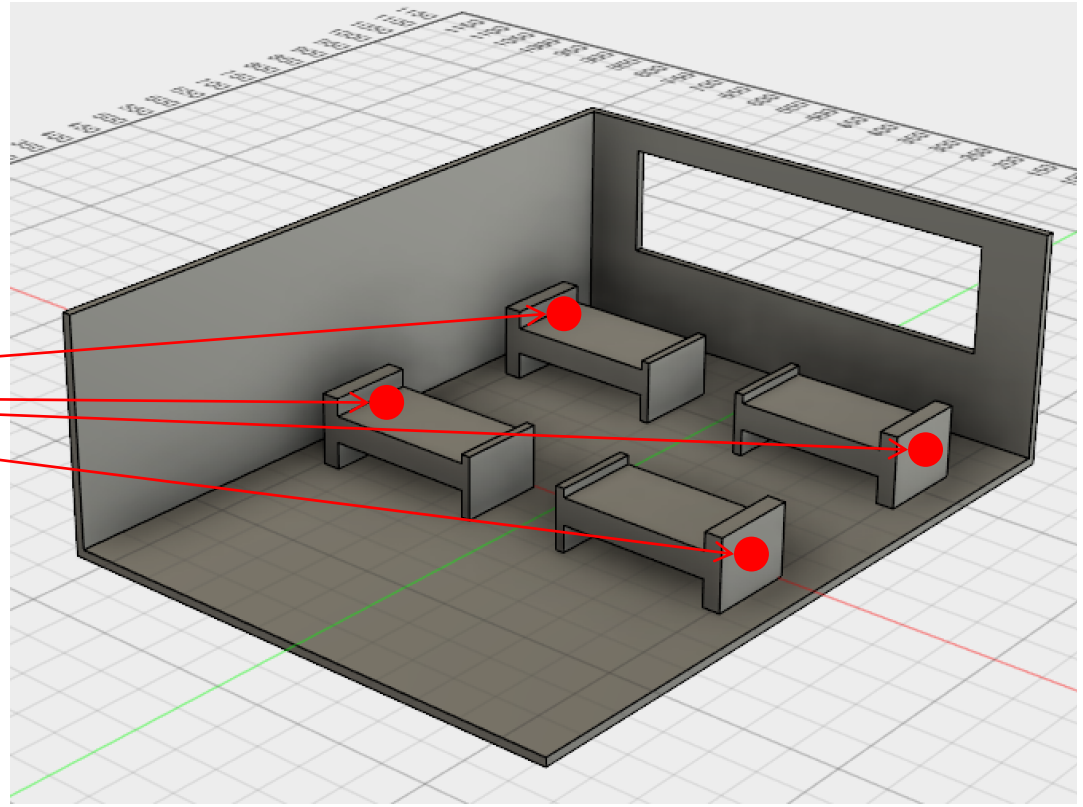
「なぜそれが必要なのか」をパートナーと協力し追求、
量産化対応、新機能、GW方式モジュール開発につなげ、
継続している。

primesap

病院での実証実験



Intel Edision
加速度/温度/湿度/照度センサー
さくらの通信モジュール



ベッド毎に入眠後の身体の動揺を計測

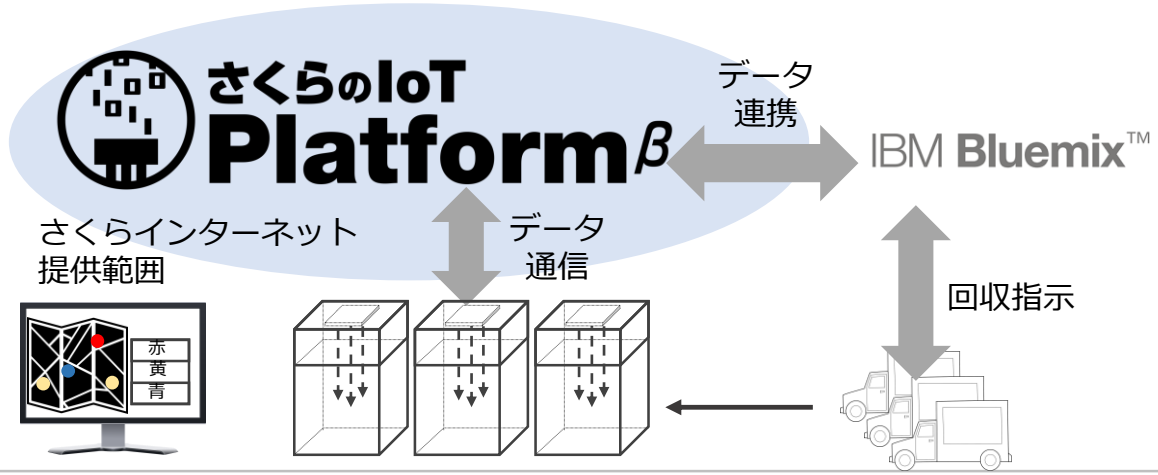
- 気圧や温度などの環境要因と安眠度の相関性について理解が進む
- 睡眠に適切な環境とさらにその個人差を把握することで環境改善に貢献

tsumug

シェアリングエコノミーを加速させる
スマートロックを中心とした
プラットフォームカンパニー

ハウステンボスとのPoC

※ハウステンボスのIoTへの取り組み
　ハウステンボス社では、IoTやAI、ロボティクスを活用しユニークな顧客体験の創造を目指しており、さくらインターネットの通信モジュールを活用した実証実験もその一環



■　**イメージ**
　パーク内に設置されたゴミ箱に計測機器を取り付け、さくらのIoTプラットフォームに送付、コグニティブコンピューティングシステムに送付し、解析する。将来的にはAIによって常に最適な回収経路を自動回収車に指示し、収集が自動的に実施される仕組みを目指す。

　パーク内ゴミ箱に、集積量を測定できるセンサーを取り付け、自動的に回収する仕組みを構築する実証実験を行うため、計測機器の通信およびデータ連携システムにさくらのIoTプラットフォームを採用

※LTE通信モジュールの他、さくらインターネットで開発中の920MHz（LoRa）および2.4GHzのゲートウェイ型モジュールも活用し、より良い利用方法についても検証を進める

御提供価格

さくらの通信モジュール

単体方式：SCM-LTE-beta

定価9,960円

100万回プラットフォームとデータ送受信可能なポイント※付き

RP=RelationPoint

さくらのIoTプラットフォーム利用時に消費されるポイント

※1つデータ(RM)をプラットフォームとやり取りすると1RP消費
※上記のみを1分間に1回行うと約2年間利用することが可能

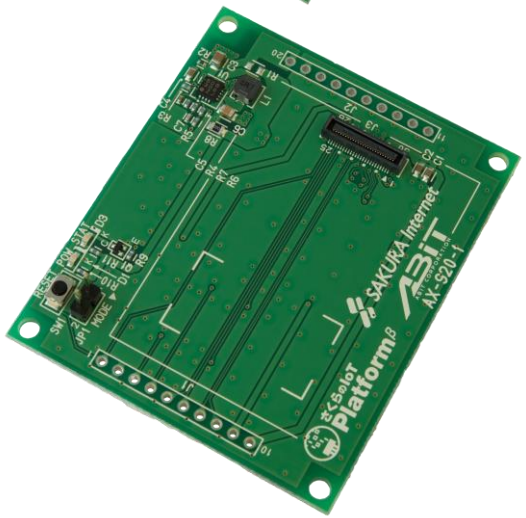
さくらの通信モジュール オプション

Arduinoシールドボード

SCO-ARD-01

Arduino マイコンボードを
利用したプロトタイプ開発に

定価8,000円



ブレイクアウトボード（検証ボード）

SCO-BB-01

その他のマイコンボードを
利用したプロトタイプ開発に

定価5,000円

さくらのIoT Platform β期間

β期間中プラットフォーム利用料無料
RP消費なし

β提供記念
キャンペーン

さくらの通信モジュール及びオプション

半額

単体方式
SCM-LTE-beta

~~9,960円~~ → **4,980円**

100万回プラットフォームと
データ送受信可能なポイント※付き

Arduino
シールドボード
SCO-ARD-01

~~8,000円~~ → **4,000円**

ブレイクアウトボード
（検証ボード）
SCO-BB-01

~~5,000円~~ → **2,500円**

ツイート

田中邦裕
@kunihirotanaka

さくらのIoTプラットフォームは、SPIやI2Cで専用通信モジュールにつなぐと、LTE回線経由でREST APIからデータを取れるサービスです。SIMが最初から入っていて2年分のLTE料金が入ってます。
iot.sakura.ad.jp/service/ #procon27

2016/10/09 16:31 場所: 三重 伊勢市

15 リツイート 22 いいね

実際に使ってみよう

さくらの IoT Platform β版ハンズオン

DAY
2016/11/27

COMPANY
さくらインターネット株式会社

DEPARTMENT
コミュニティマネージャー

NAME
法林 浩之

ハンズオンの資料に沿って実際に使っている様子を紹介

SAKURA Internet

Webサービス連携
（さくらのクラウド）

マイコンおよび
プログラムの構築

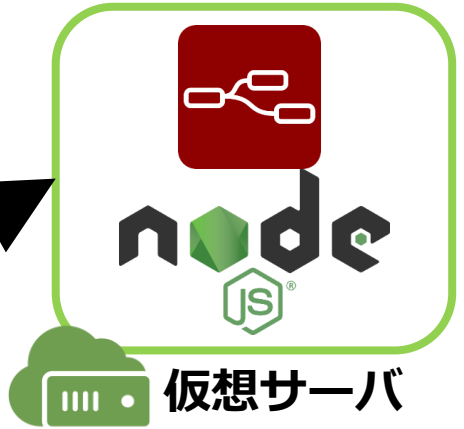
温湿度センサ
（HDC1000）

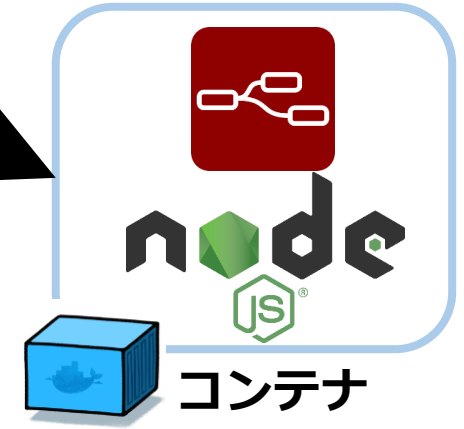
マイコン
（Arduino Uno）

さくらの通信
モジュール

さくらのIoT
Platformの設定

さくらのIoT
Platformβ





Webサービス連携
（Arukas）

1. マイコンおよびプログラムの構築

- マイコン（Arduino）による開発環境の準備
- 温湿度センサおよびさくらの通信モジュールの繋ぎ込み
- 試験用プログラムの流し込み

2. さくらのIoT Platformの設定

- プロジェクトの作成
- さくらの通信モジュールの登録
- 連携サービスの設定

3. Webへのデータ連携（さくらのクラウド）

- Node-REDサーバの作成
- WebSocketを利用したデータ連携フロー作成
- Zabbixサーバへのデータ連携とグラフ化※ハンズオン対象外

マイコンおよび
プログラムの構築

今回のハンズオンの流れ

SAKURA Internet

マイコンおよび
プログラムの構築

温湿度センサ
（HDC1000）

マイコン
（Arduino Uno）

さくらの通信
モジュール

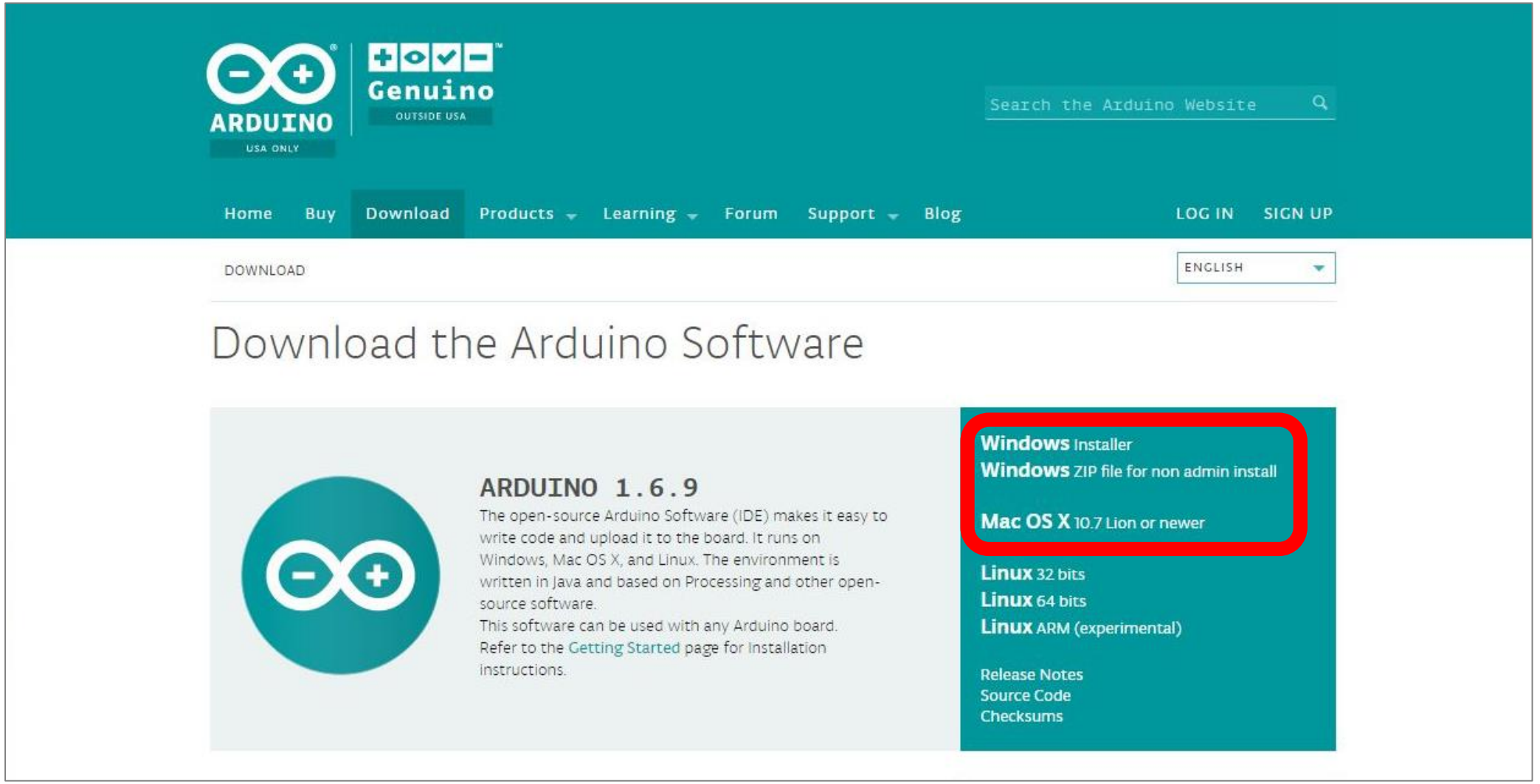
さくらのIoT
Platformの設定

さくらのIoT
Platformβ

Webサービス連携
（さくらのクラウド）

Webサービス連携
（Arukas）

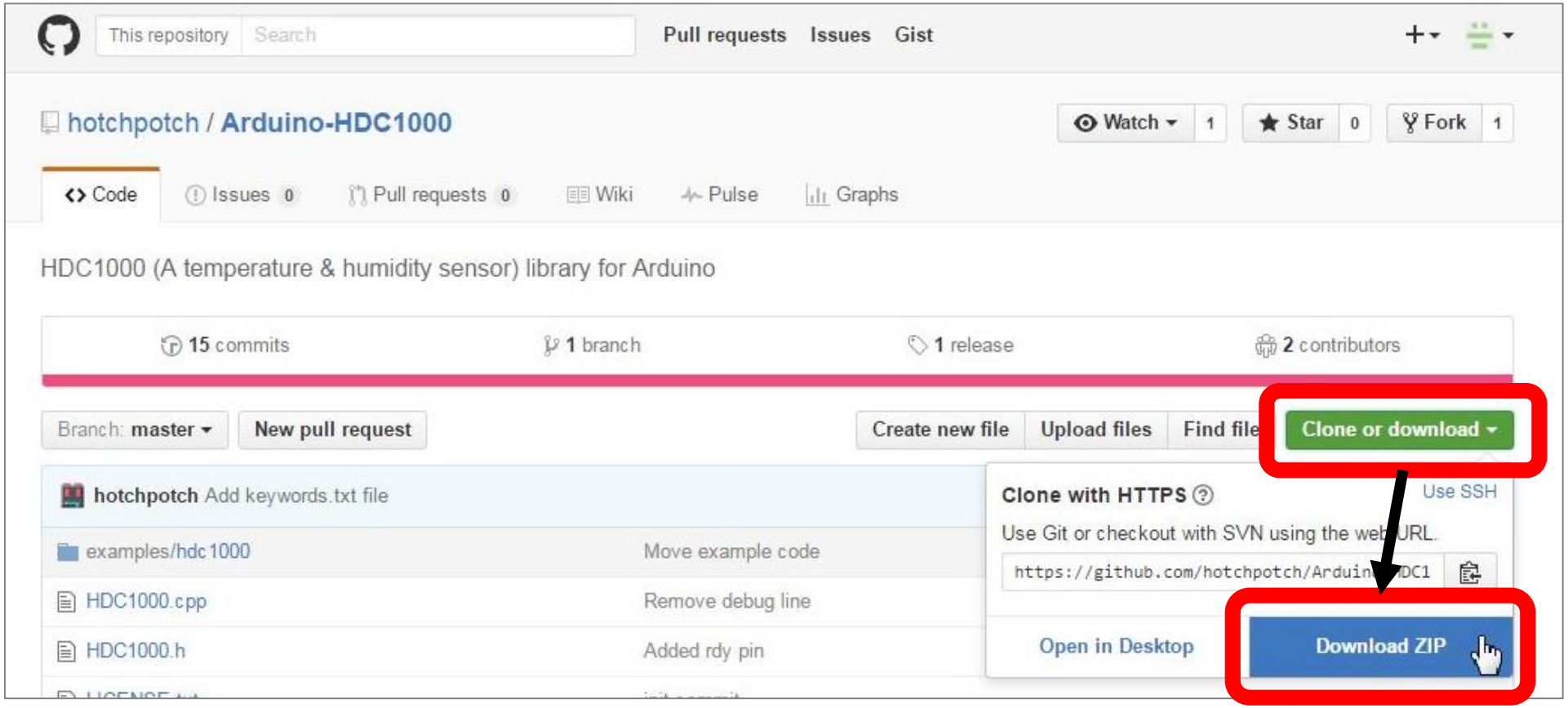
https://www.arduino.cc/en/Main/Software から開発環境（Arduino IDE）を入手します。
2016/10/18時点での最新版は【1.6.12】となります。



温湿度センサ HDC1000 ライブラリ

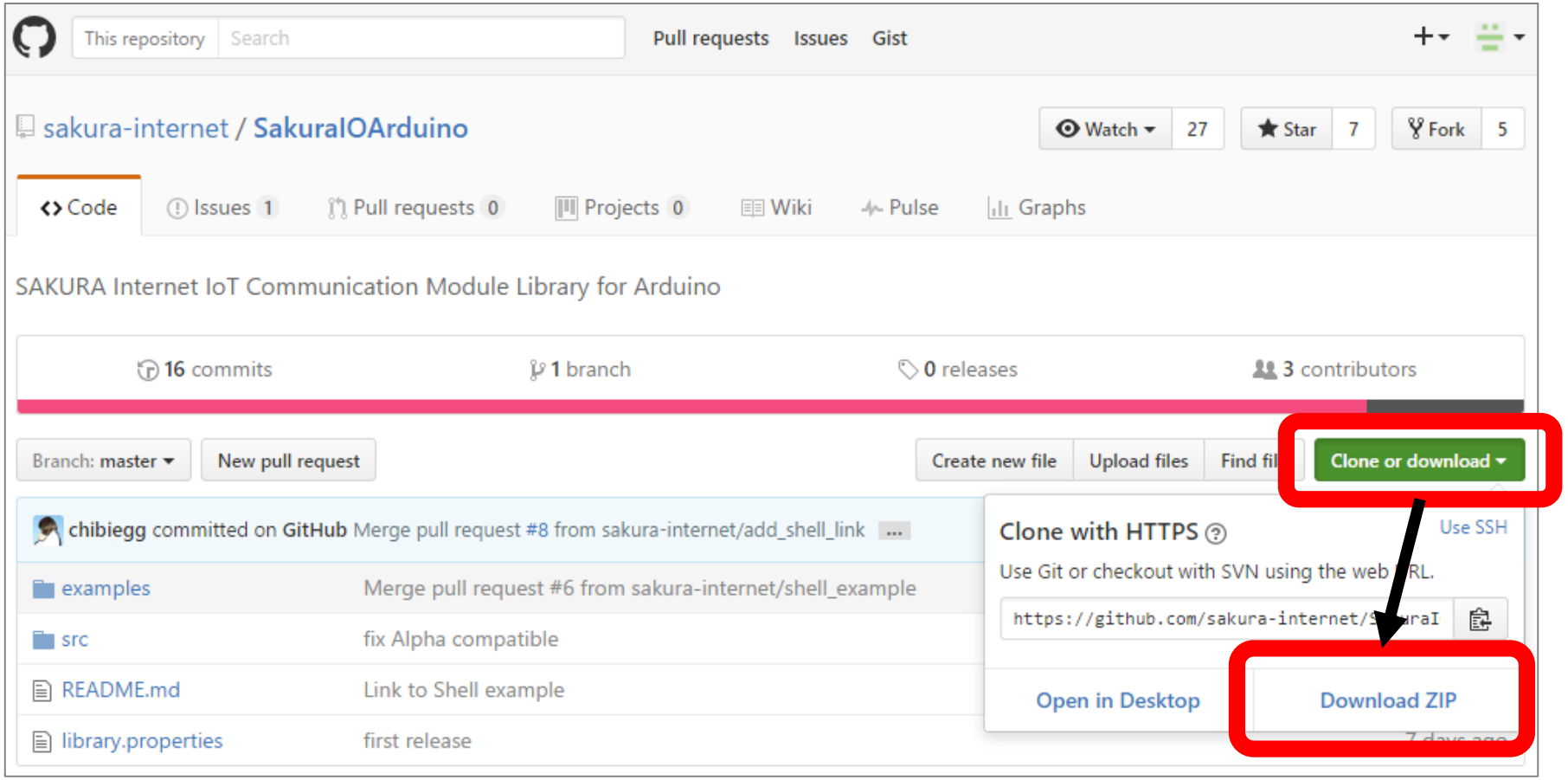
Googleにて「github hdc1000」を検索し、ライブラリを入手します。
https://github.com/**hotchpotch**/Arduino-HDC1000

入手したZipファイルは**解凍**し、「Arduino-HDC1000-master」フォルダごと以下へ格納します。
Windows　　C:¥Users¥《ユーザ名》¥Documents¥Arduino¥libraries
Mac　　書類¥Arduino¥libraries



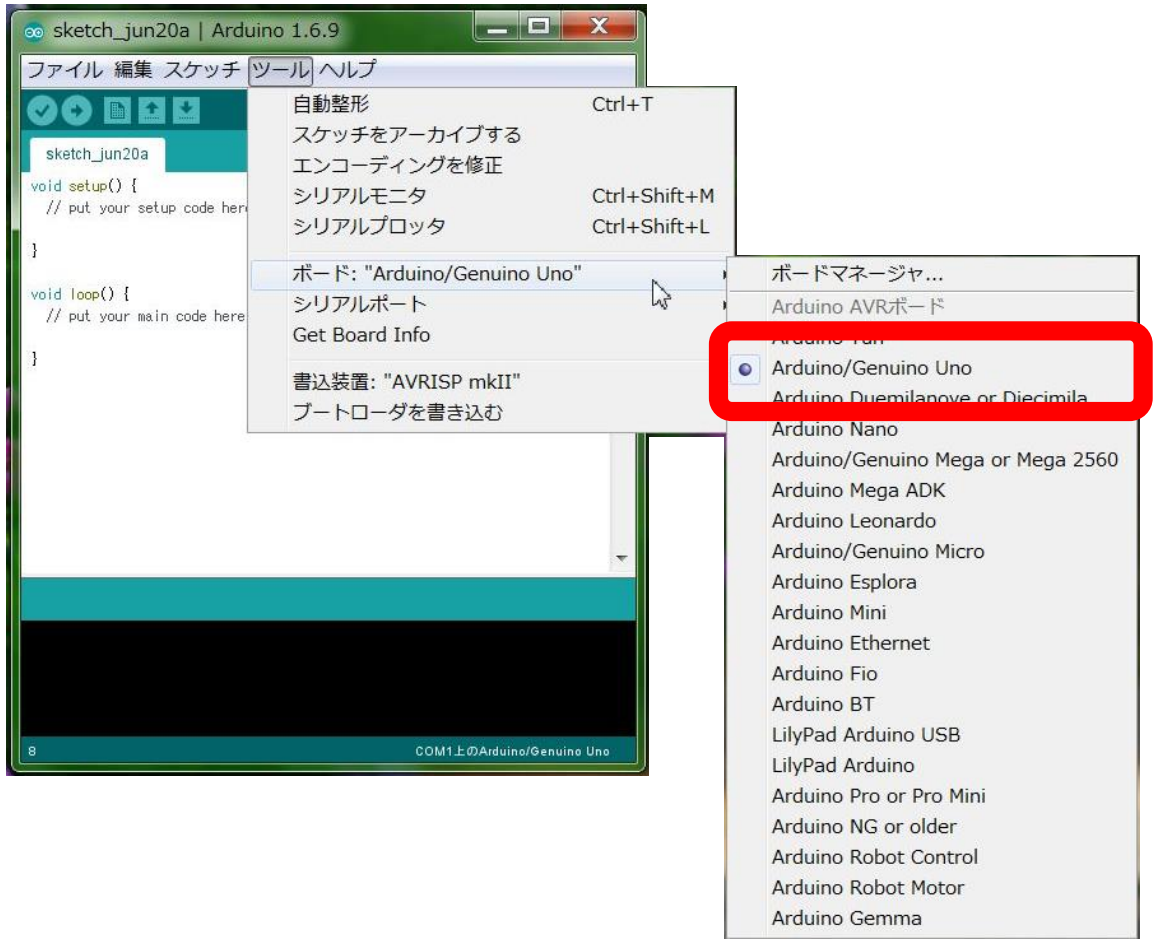
Googleにて「github sakuraioarduino」を検索し、ライブラリを入手します。
https://github.com/sakura-internet/SakuraIOArduino

入手したZipファイルは**解凍**し、「SakuraIOArduino-master」フォルダごと以下へ格納します。
Windows　C:¥Users¥《ユーザ名》¥Documents¥Arduino¥libraries
Mac　書類¥Arduino¥libraries

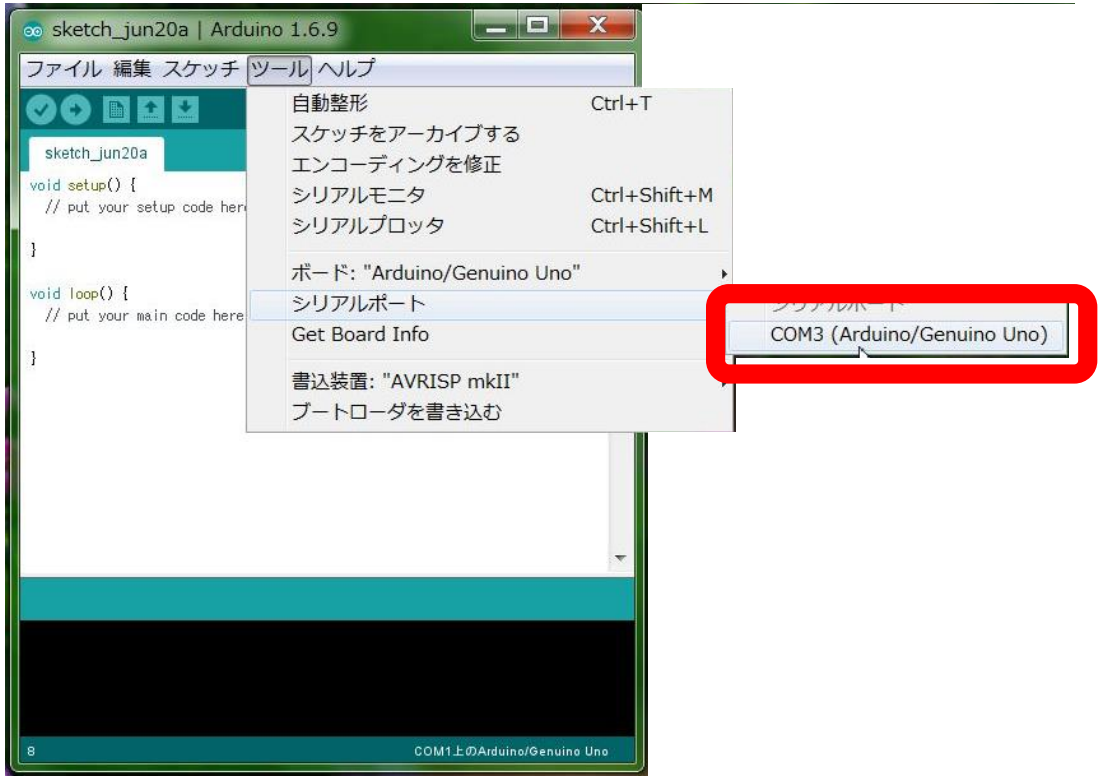


Arduino IDEが起動したら、Arduino本体をPCに接続します。
ボードは【Arduino/Genuino Uno】、シリアルポートは出てきたCOMポートを選択します。
Macの場合はシリアルポート内の「～～～（Arduino/Genuino Uno）」となるものを選択します。

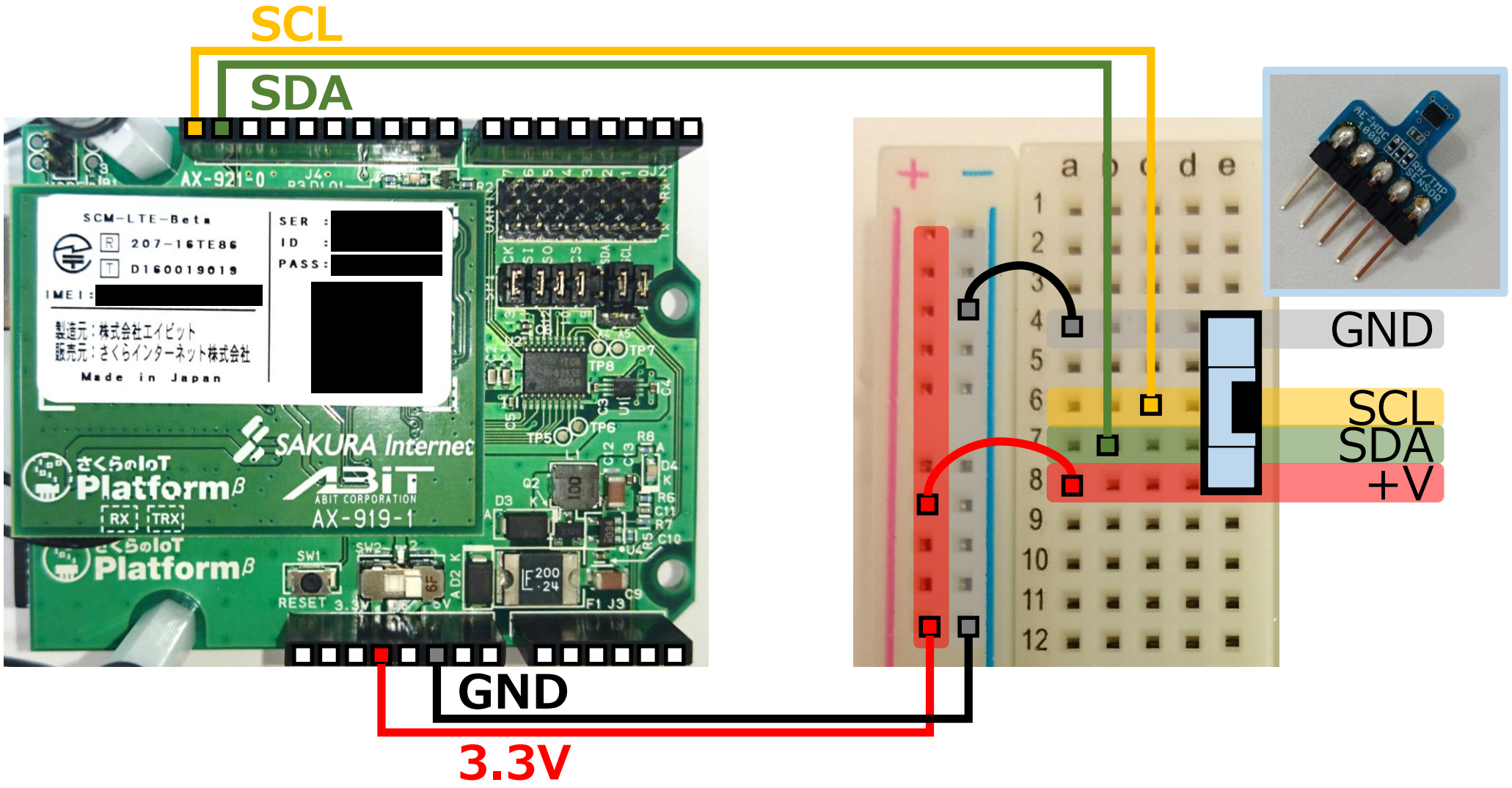
ボードの選択

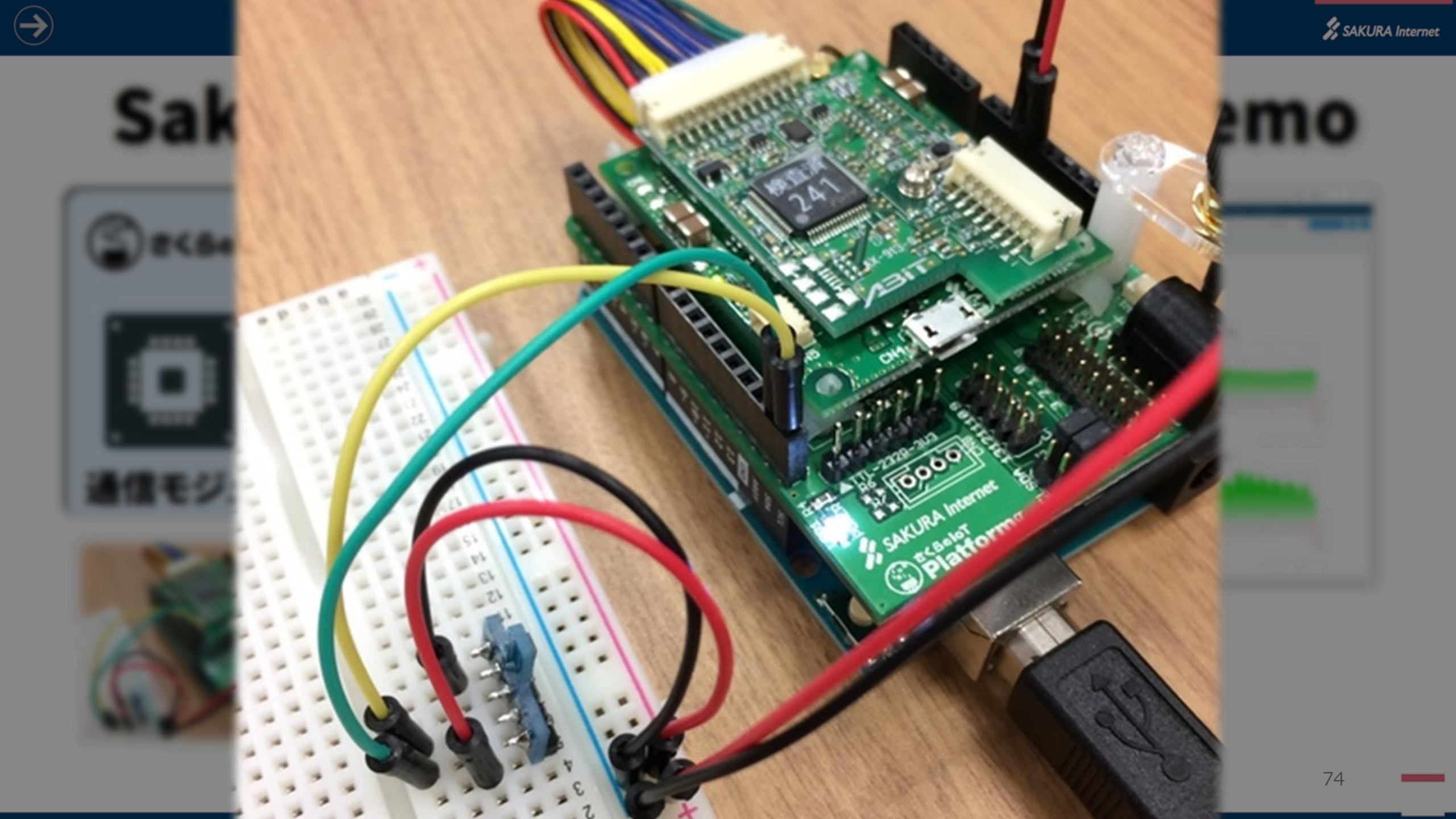


シリアルポートの選択



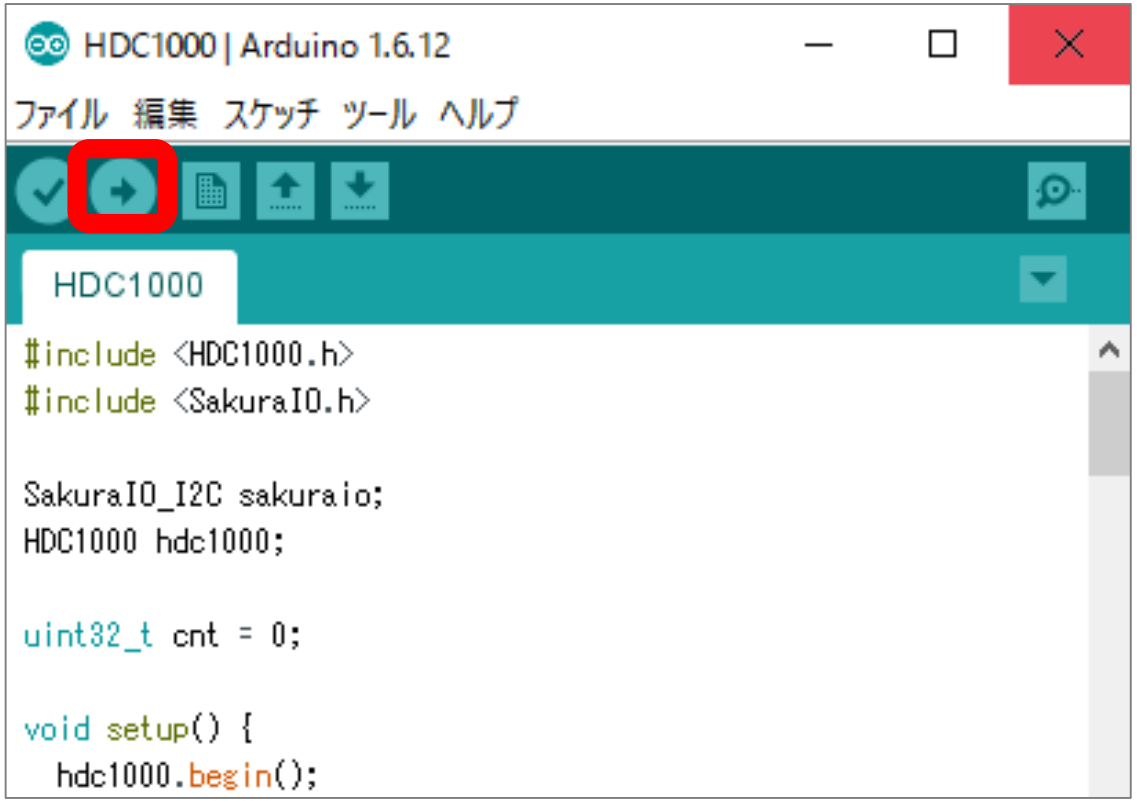
結線を行うため、ArduinoをPCから外します。
その後、図に従ってジャンパーコードを接続します。

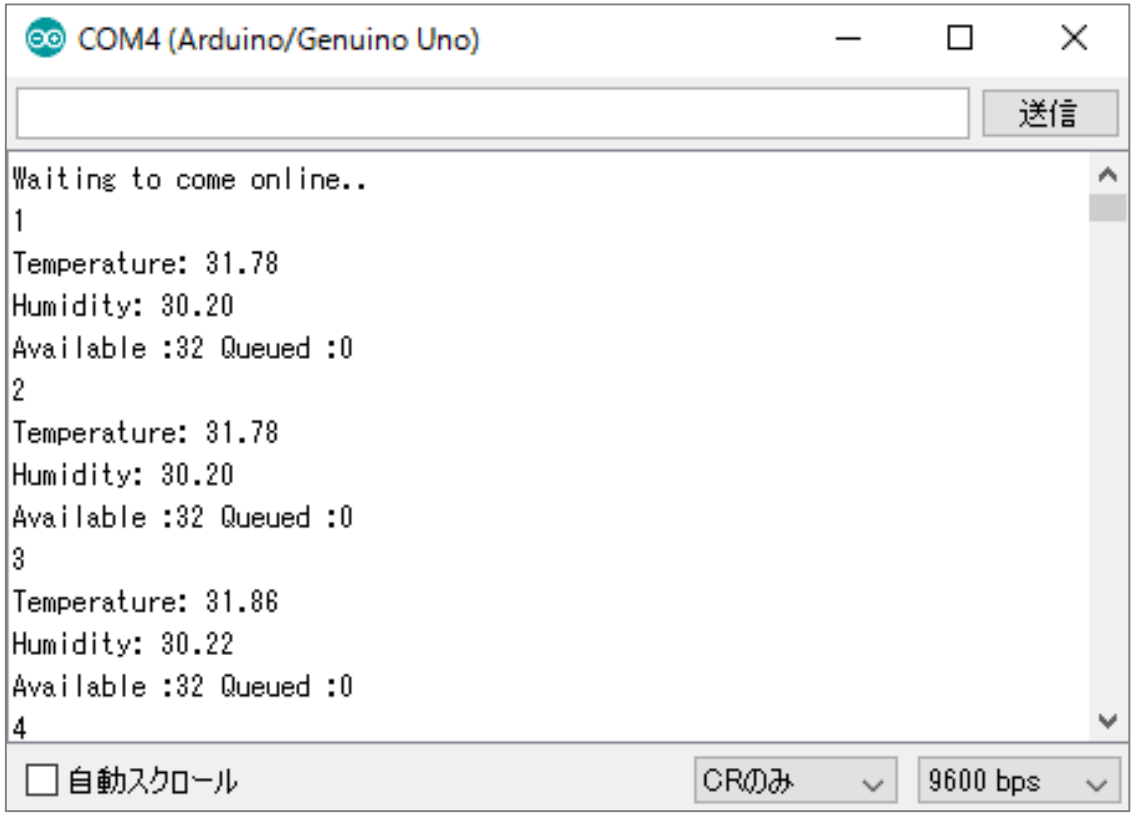



SAKURA Internet
SAKURA Internet
Platform

試験用プログラムの流し込み

[スケッチの例]→[SakuraIO]→[HDC1000] →【→】ボタンをクリックします。
[ツール]→[シリアルモニタ]より「Waiting to come online」表記の後、カウント値、Temperature、Humidityに加え、Available（キューイング可能なチャンネル数）とQueued（キューで送信待ちになっているチャンネル数）が表示されることを確認します。





さくらのIoT Platform の設定

今回のハンズオンの流れ

SAKURA Internet

Webサービス連携
（さくらのクラウド）

マイコンおよび
プログラムの構築

さくらのIoT
Platformの設定

温湿度センサ
（HDC1000）

マイコン
（Arduino Uno）

さくらの通信
モジュール

さくらのIoT
Platformβ

仮想サーバ

コンテナ

Webサービス連携
（Arukas）

コントロールパネルへのログイン

Googleにて「さくらインターネット iot 開発者」を検索し、開発者向けページから
コントロールパネル（https://secure.sakura.ad.jp/iot-alpha/）にログインします。

プロジェクトを作成し、通信モジュールを登録し、連携サービスを設定します。

作成した連携サービスで、シリアルポートで表示される情報（温度/湿度/シリアル値）と
同様の情報が画面上で受信データとして表示されていることを確認します。

送信されたデータの
タイムスタンプ

データを送信した
通信モジュールのID

データが格納された
チャンネル番号

データの型

送信された値

受信データ

時刻	モジュール	チャンネル	型	値	
2016-11-16T08:26:36.592526903Z	[redacted]	2	I	22	→カウント値
2016-11-16T08:26:36.592526903Z	[redacted]	1	f	36.120605	→湿度
2016-11-16T08:26:36.592526903Z	[redacted]	0	f	28.995056	→温度
2016-11-16T08:26:35.51272341Z	[redacted]	2	I	21	
2016-11-16T08:26:35.51272341Z	[redacted]	1	f	36.120605	
2016-11-16T08:26:35.51272341Z	[redacted]	0	f	28.964844	

単一メッセージで
送信された値は
同一時刻で表示

Webへのデータ連携
（Node-RED編）

今回のハンズオンの流れ

SAKURA Internet

Webサービス連携
（さくらのクラウド）

マイコンおよび
プログラムの構築

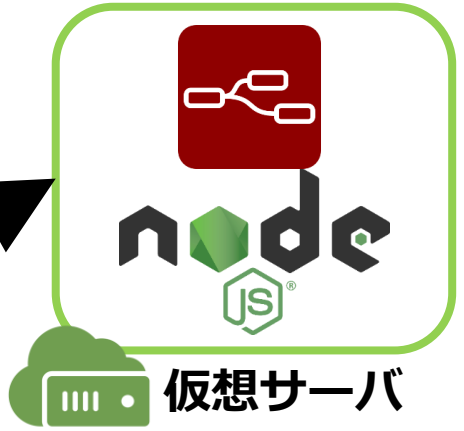
さくらのIoT
Platformの設定

温湿度センサ
（HDC1000）

マイコン
（Arduino Uno）

さくらの通信
モジュール

さくらのIoT
Platformβ



仮想サーバ

コンテナ

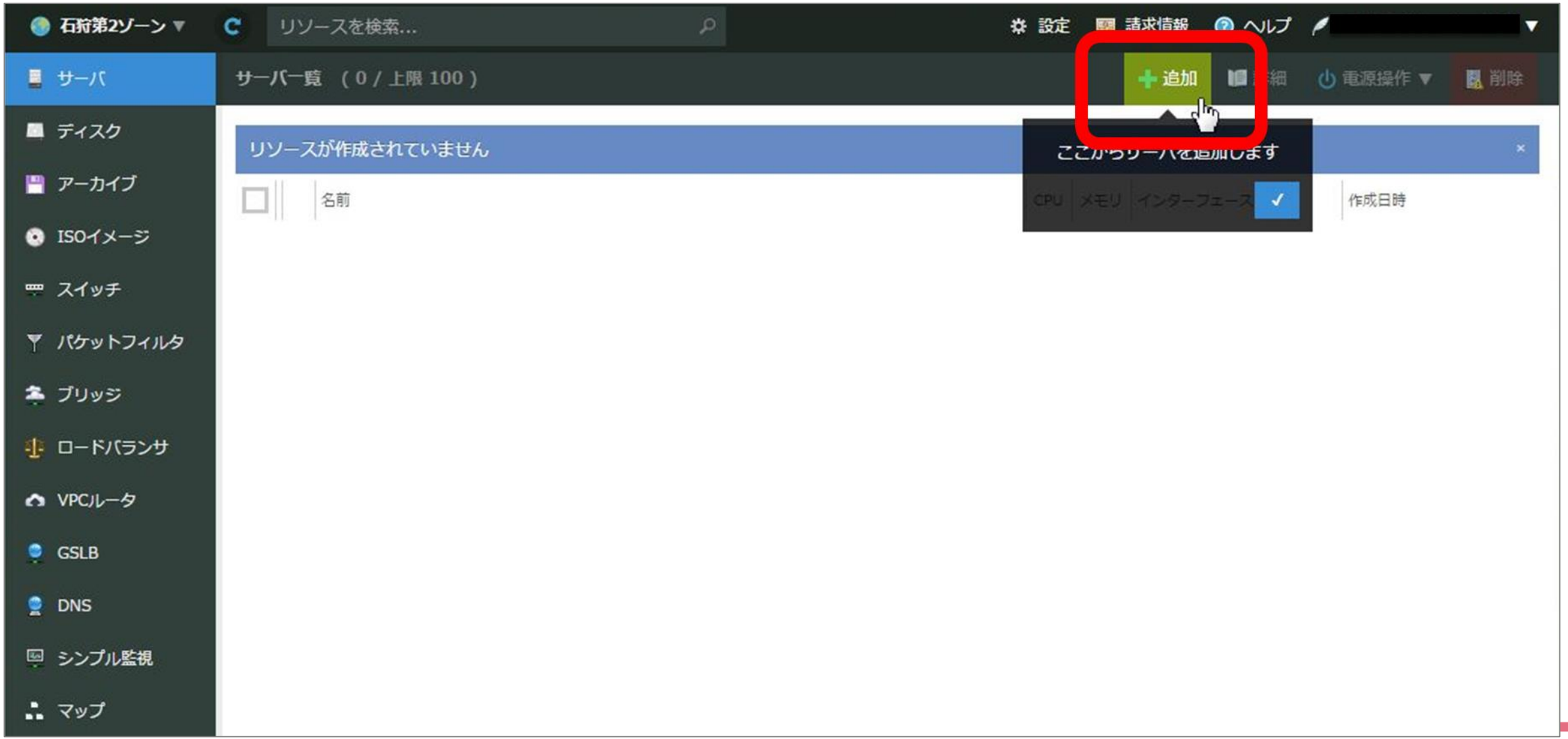
Webサービス連携
（Arukas）

さくらのクラウド コントロールパネル ログイン

以下URLより、「さくらのクラウド」のコントロールパネルにログインします。
「https://secure.sakura.ad.jp/cloud/」

サーバの作成

左側のペインのサーバを選択し、右上の【追加】ボタンをクリックします。
はじめはサーバ追加の案内が出る場合があります。



作業概要

- Gitの入手
- NVM(Node Version Manager)の入手と設定
- Node.jsの入手
- Node-REDの入手と設定と起動

まずはyumコマンドでGitのパッケージをインストールします。

```
------------------------------------------------------------
#Gitのインストール
------------------------------------------------------------
[root@test ~]# yum -y install git
Loaded plugins: fastestmirror, priorities, security
Setting up Install Process
Determining fastest mirrors
epel/metalink                                      | 5.2 kB     00:00
 * elrepo: ftp.ne.jp
 * epel: www.ftp.ne.jp

～中略～

Updated:
  git.x86_64 0:1.7.1-4.el6_7.1

Dependency Updated:
  perl-Git.noarch 0:1.7.1-4.el6_7.1

Complete!
```

次にGitを使用してNode Version Manager(NVM)をインストールします。

```
------------------------------------------------------------
#Node Version Manager(NVM)のインストール
------------------------------------------------------------
[root@test ~]# git clone https://github.com/creationix/nvm.git ~/.nvm
Initialized empty Git repository in /root/.nvm/.git/
remote: Counting objects: 4732, done.
remote: Total 4732 (delta 0), reused 0 (delta 0), pack-reused 4731
Receiving objects: 100% (4732/4732), 1.27 MiB | 544 KiB/s, done.
Resolving deltas: 100% (2814/2814), done.

[root@test ~]# source ~/.nvm/nvm.sh

[root@test ~]# nvm help

Node Version Manager

～中略～

to remove, delete, or uninstall nvm - just remove the `$NVM_DIR` folder (usually `~/.nvm`)
```

コンソールでの環境のセットアップ

NVMを使用し、Node-REDが稼動するためのNode.jsをインストールします。

```
------------------------------------------------------------
#Node.jsのインストール
------------------------------------------------------------
[root@test ~]# nvm ls-remote
        v0.1.14
        v0.1.15

～中略～

         v6.2.1
         v6.2.2

[root@test ~]# nvm install v6.2.2
Downloading https://nodejs.org/dist/v6.2.2/node-v6.2.2-linux-x64.tar.xz...
######################################################################## 100.0%
Now using node v6.2.2 (npm v3.9.5)
Creating default alias: default -> v6.2.2

[root@test ~]# node -v
v6.2.2
```

コンソールでの環境のセットアップ

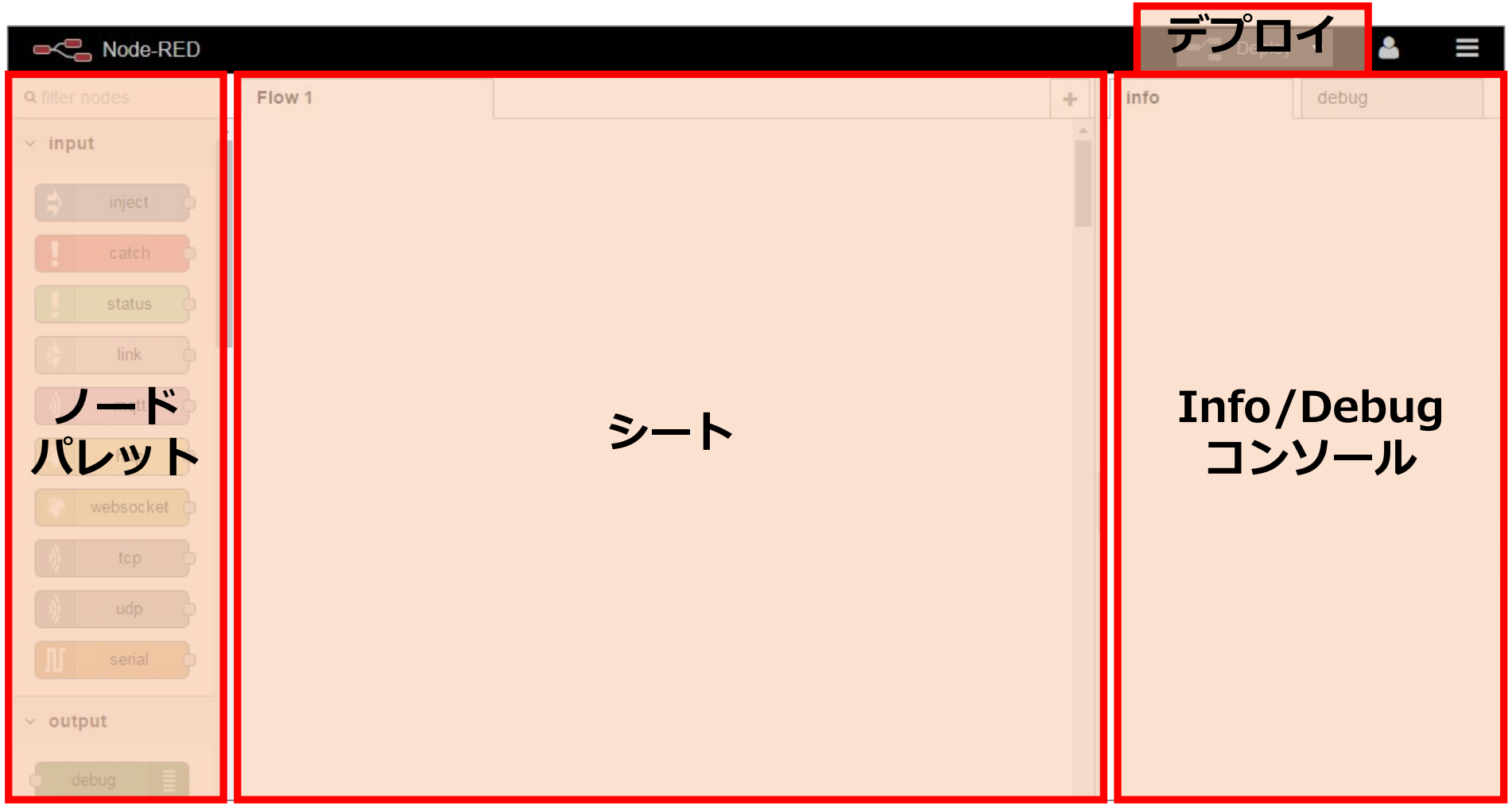
Node-REDをインストールし、起動します。以降はWebブラウザで作業します。

```
------------------------------------------------------------
#Node-Redのインストール
------------------------------------------------------------
[root@test ~]# npm install -g node-red
npm WARN deprecated i18next-client@1.10.3: you can use npm install i18next from version 2.0.0
/root/.nvm/versions/node/v6.2.2/bin/node-red ->

～中略～

`-- xmlbuilder@4.2.1
    `-- lodash@4.13.1

[root@test ~]# node-red

Welcome to Node-RED

～中略～

21 Jun 14:29:05 - [info] Started flows
21 Jun 14:29:05 - [info] Server now running at http://127.0.0.1:1880/
```

ここのIPアドレスをサーバのグローバルIPアドレスに変更してブラウザでアクセス

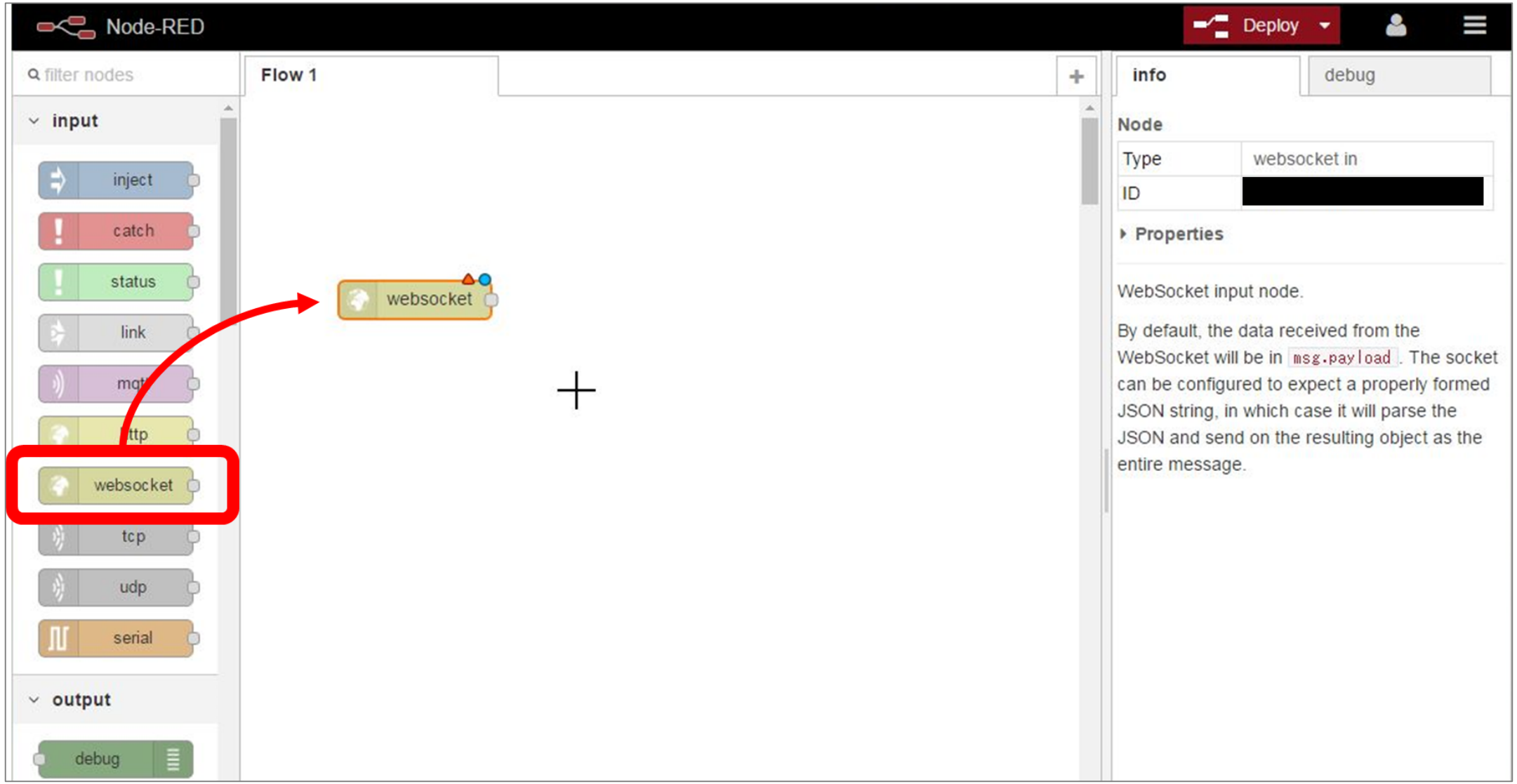
Node-RED 初期画面

Node-REDは「ノード」と呼ばれる機能の固まりをシート上で組み合わせ、
ひとつの「フロー」にすることで、ほとんどプログラミングを知らない人でも
プログラムを構築することができるツールとなります。

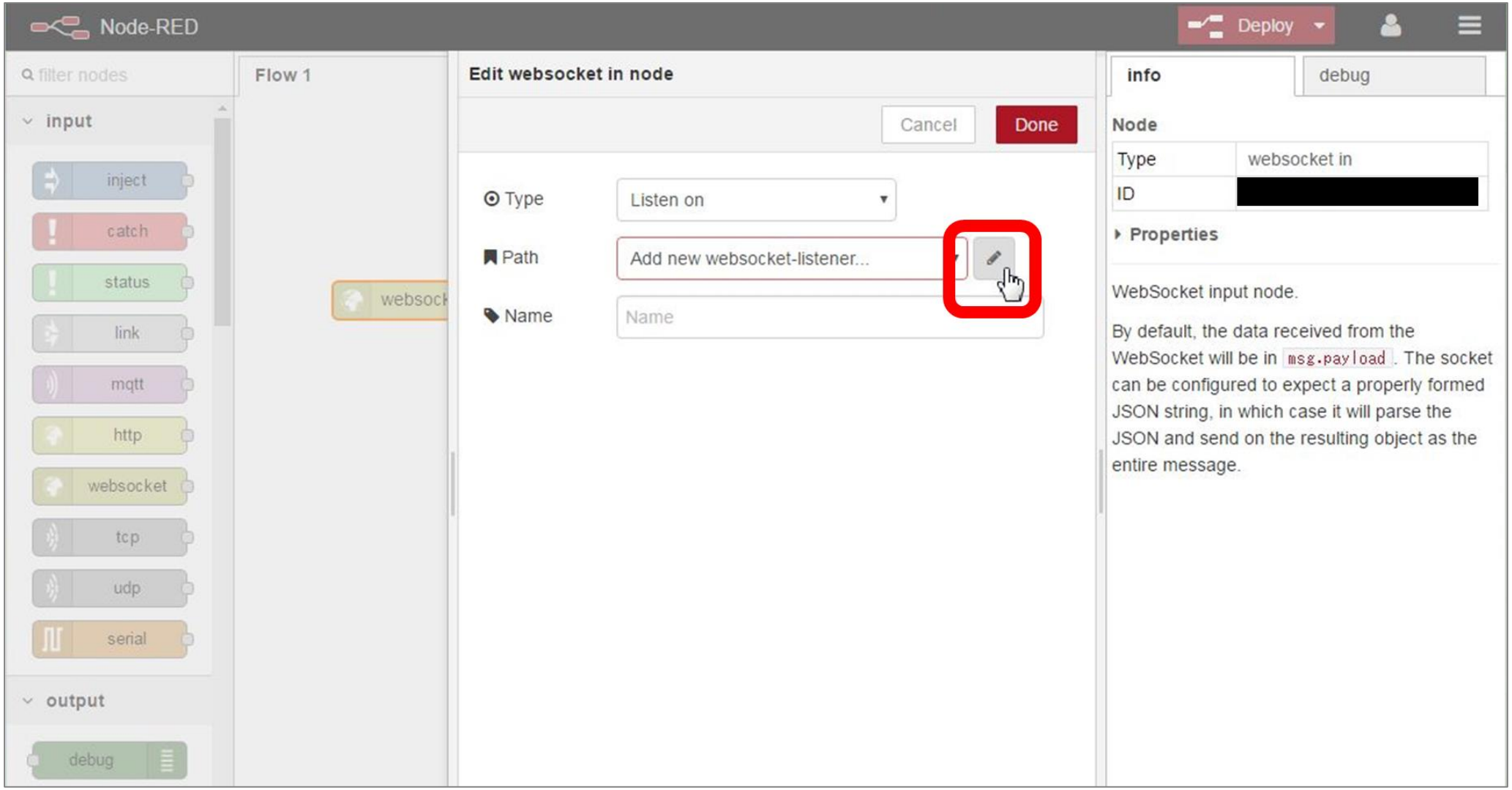


Node-RED WebSocketノードの作成

まずはWebSocketからのデータを受け取るノードを追加します。
ノードパレットから「websocket」ノードをシートにドラッグ&ドロップします。



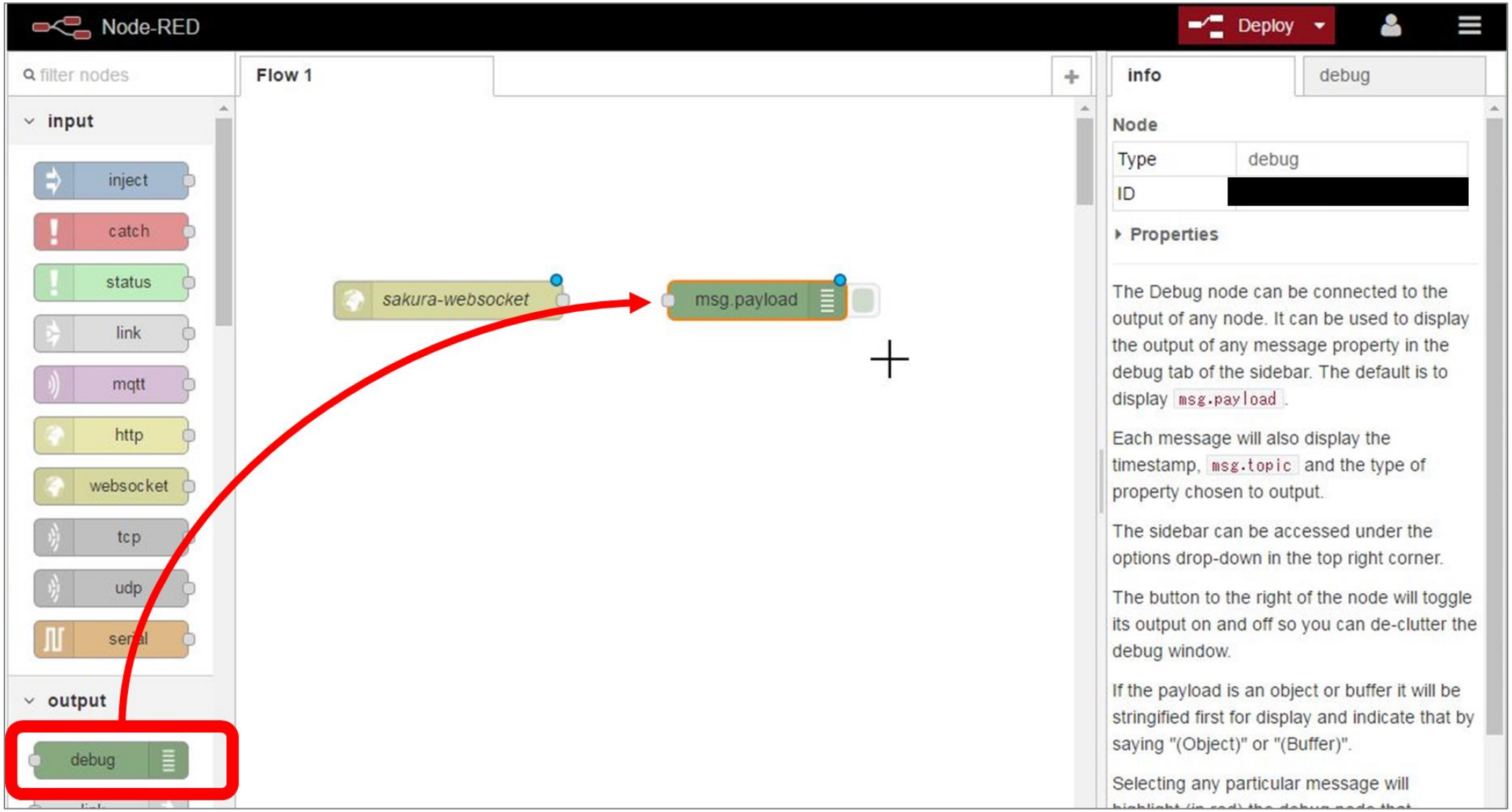
ドラッグ&ドロップされたWebSocketノードをWクリックし、設定画面に移ります。
まずはPathの行にある鉛筆マークをクリックします。



指定するPathの値は、コンパネの連携サービスで確認できる赤枠部分となります。
赤枠部分の情報をコピーして、WebSocketノードのPath部分にペーストします。

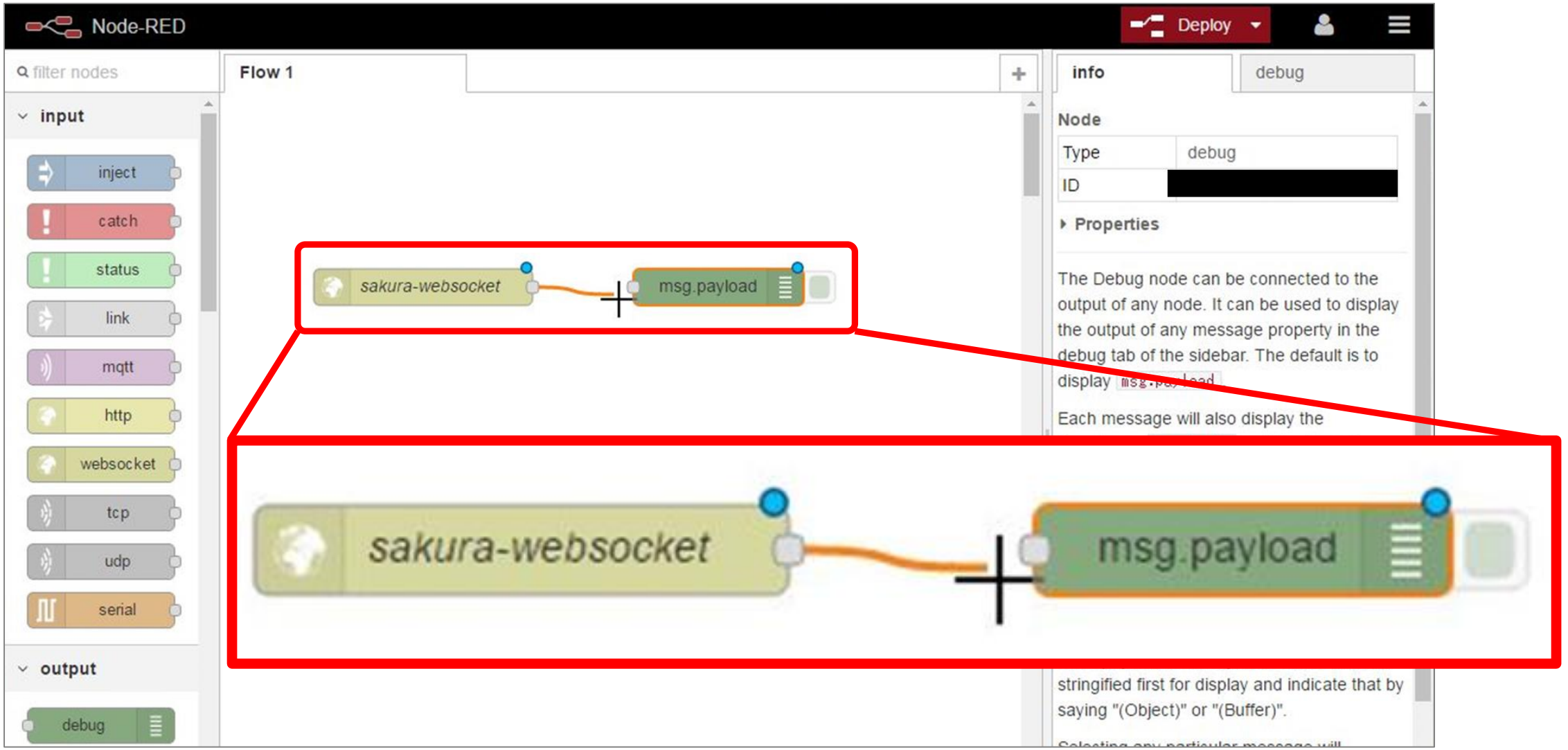
Node-RED Debugノードの作成

次に、ノードパレットから「debug」ノードをシートにドラッグ&ドロップします。
Debugノードは自動で「msg.payload」に名前が変わり、特に設定は不要です。



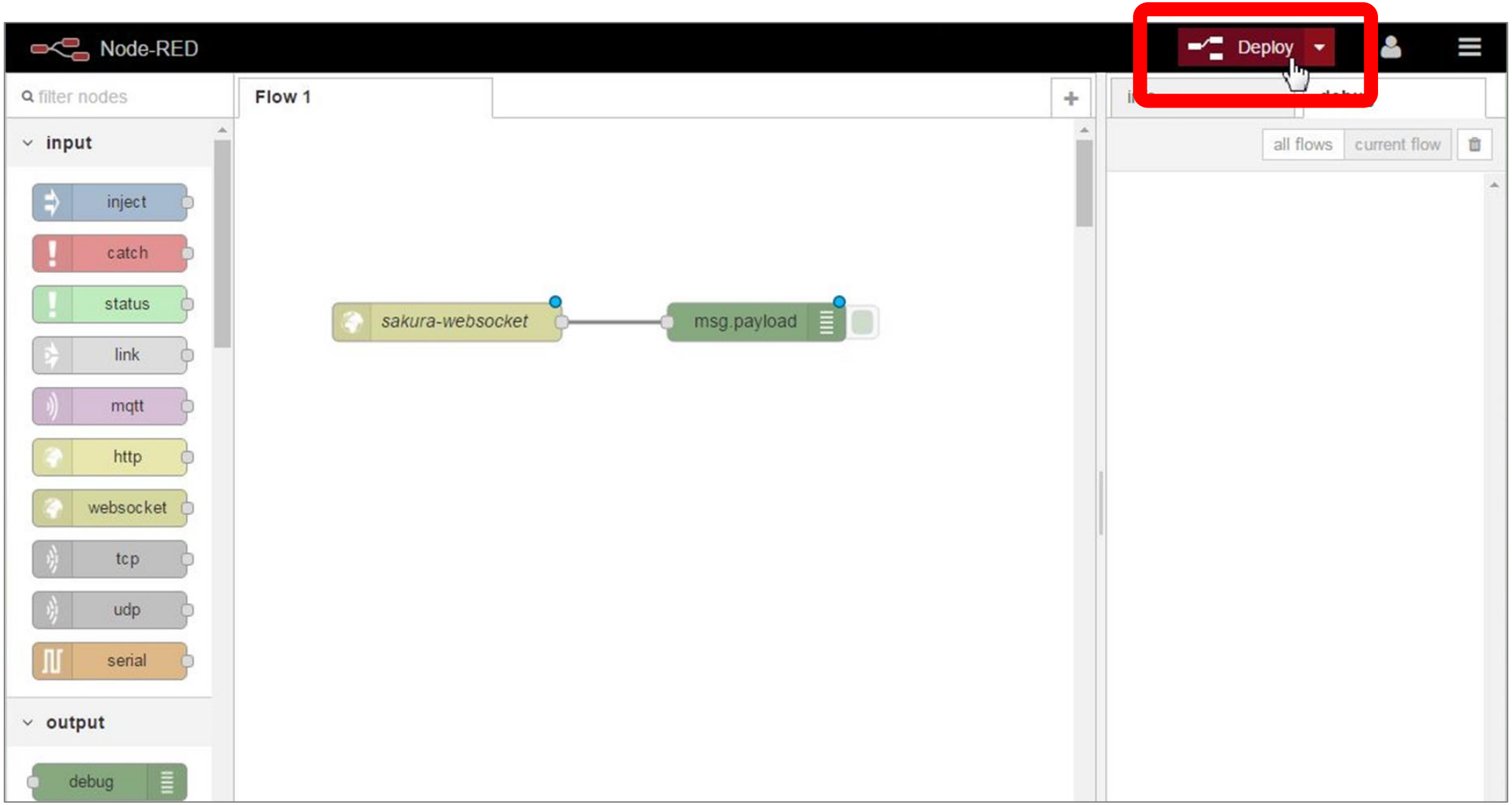
Node-RED ノード間の接続

各ノードでの動作を処理として繋げるために、WebSocketノードの右端とDebugノードの左端をドラッグ&ドロップで線を繋ぎます。

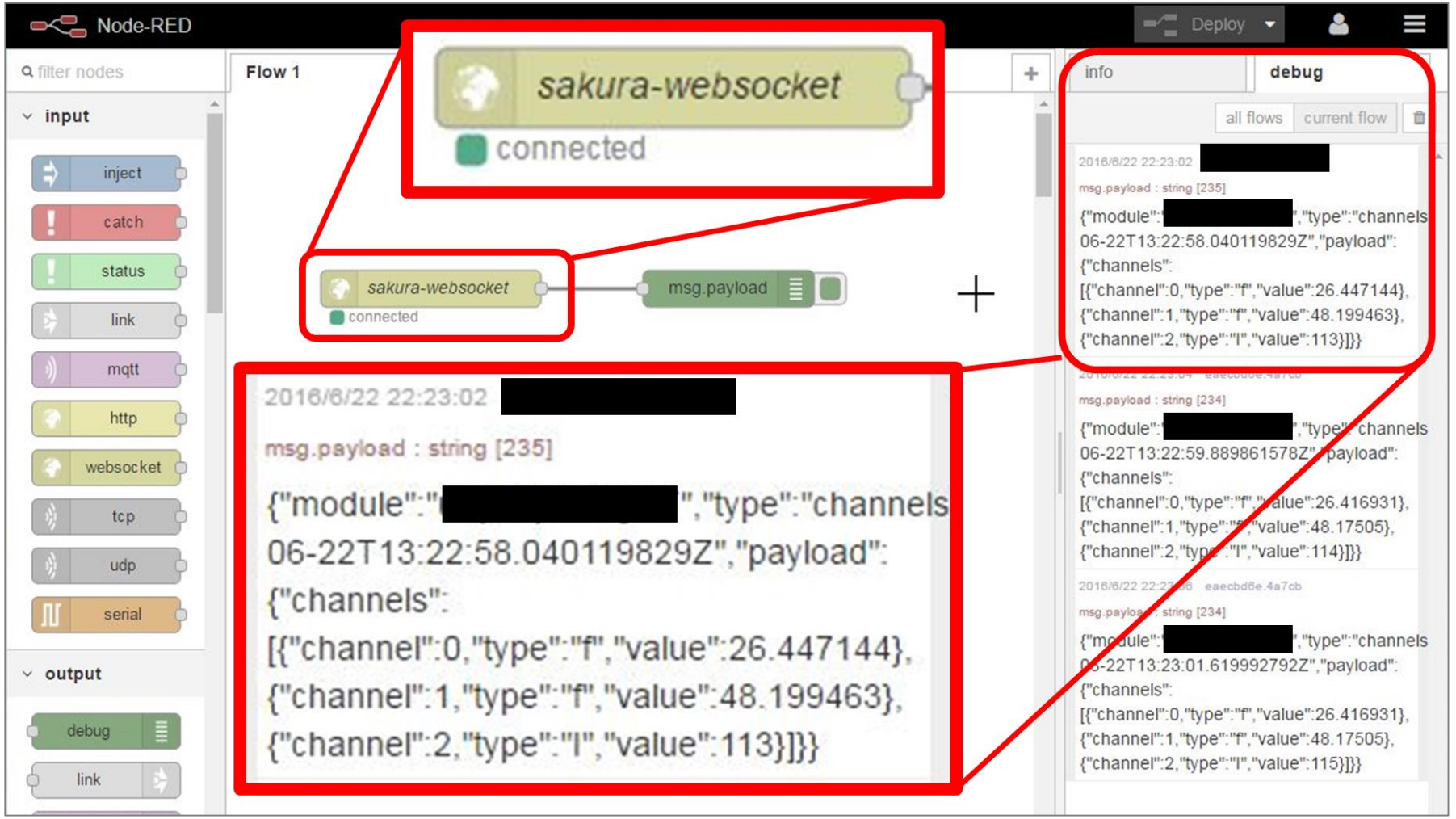


Node-RED フローのデプロイ

各ノードを接続し、準備が完了したら、右上部の【Deploy】をクリックします。
デプロイが完了するとDeployボタンがグレーアウトされ、設定した内容を元に処理が開始されます。

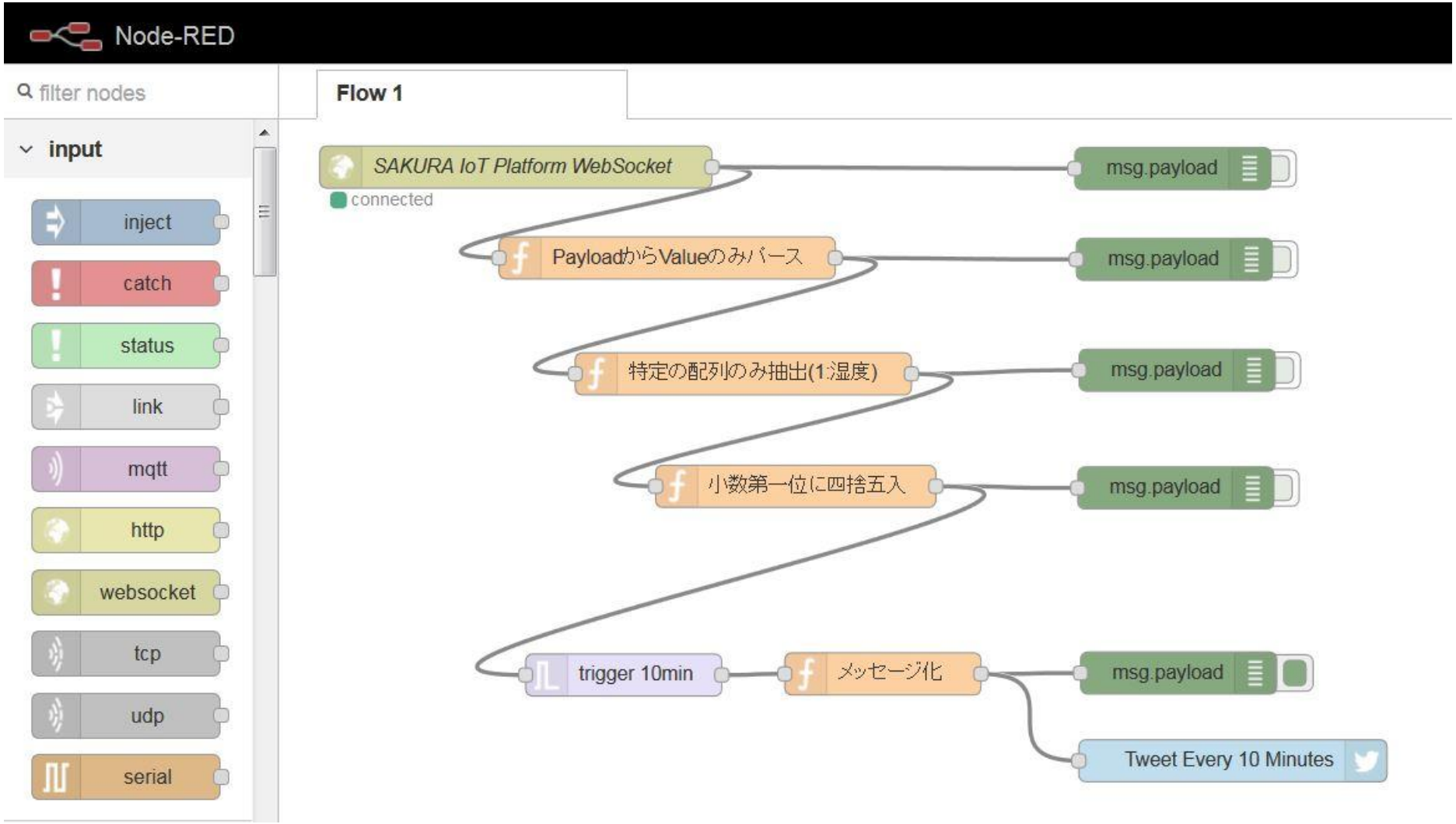


フローに問題がない場合、Websocketノード下部に「connected」と表示され、
コンソールのdebug内にプラットフォームから取得したJSONデータを確認できます。
Debugノード右端の緑マークをクリックするとdebugへの表示が停止されます。



取得した温湿度をTwitterに投稿

Websocketでデータを入手 → 温度と湿度を抽出 → 小数第一位で四捨五入
→ 10分ごとにトリガー発生 → メッセージ作成 → Twitterに投稿



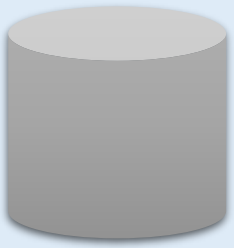
展示ブースにて実演中


ツイート　ツイートと返信
法林浩之(バースト用) @hourin_burst · 11月16日
さくらインターネットのハンズオンで湿度情報を取得中！ただ今の現地湿度は56度だよ。β版もよろしくね！ #さくらのIoTPlatform #さくらインターネット
法林浩之(バースト用) @hourin_burst · 11月16日
さくらインターネットのハンズオンで湿度情報を取得中！ただ今の現地湿度は53.4度だよ。β版もよろしくね！ #さくらのIoTPlatform #さくらインターネット

Webへのデータ連携
（Zabbix編）

さくらのIoT Platformα

3G/LTE
閉域網

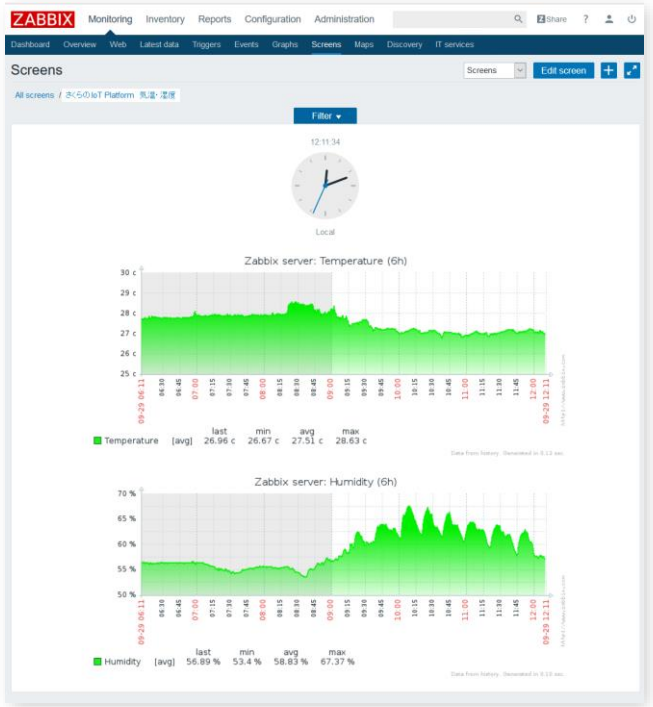


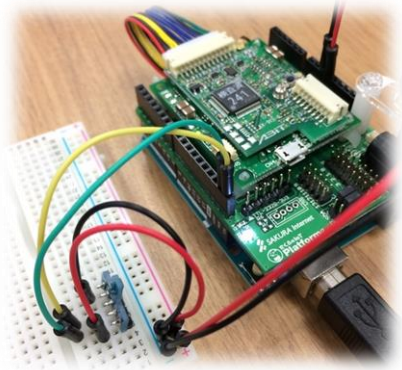
通信モジュール

WebSocket

さくらのクラウド
sakura cloud

Linux (CentOS 7.2)
http://zabbix.sakura-pr.jp/
(59.106.222.91)





Arduino
温湿度センサ

WebSocket 受信スクリプト(Perl)
↓
zabbix_sender で Zabbix に送信
↓
Zabbix でデータ収集・グラフ化・アクション

- サーバの作成
- Zabbixサーバの構築
- Zabbixにおける監視の設定
- さくらのIoT Platformからのデータ取得プログラムの設置

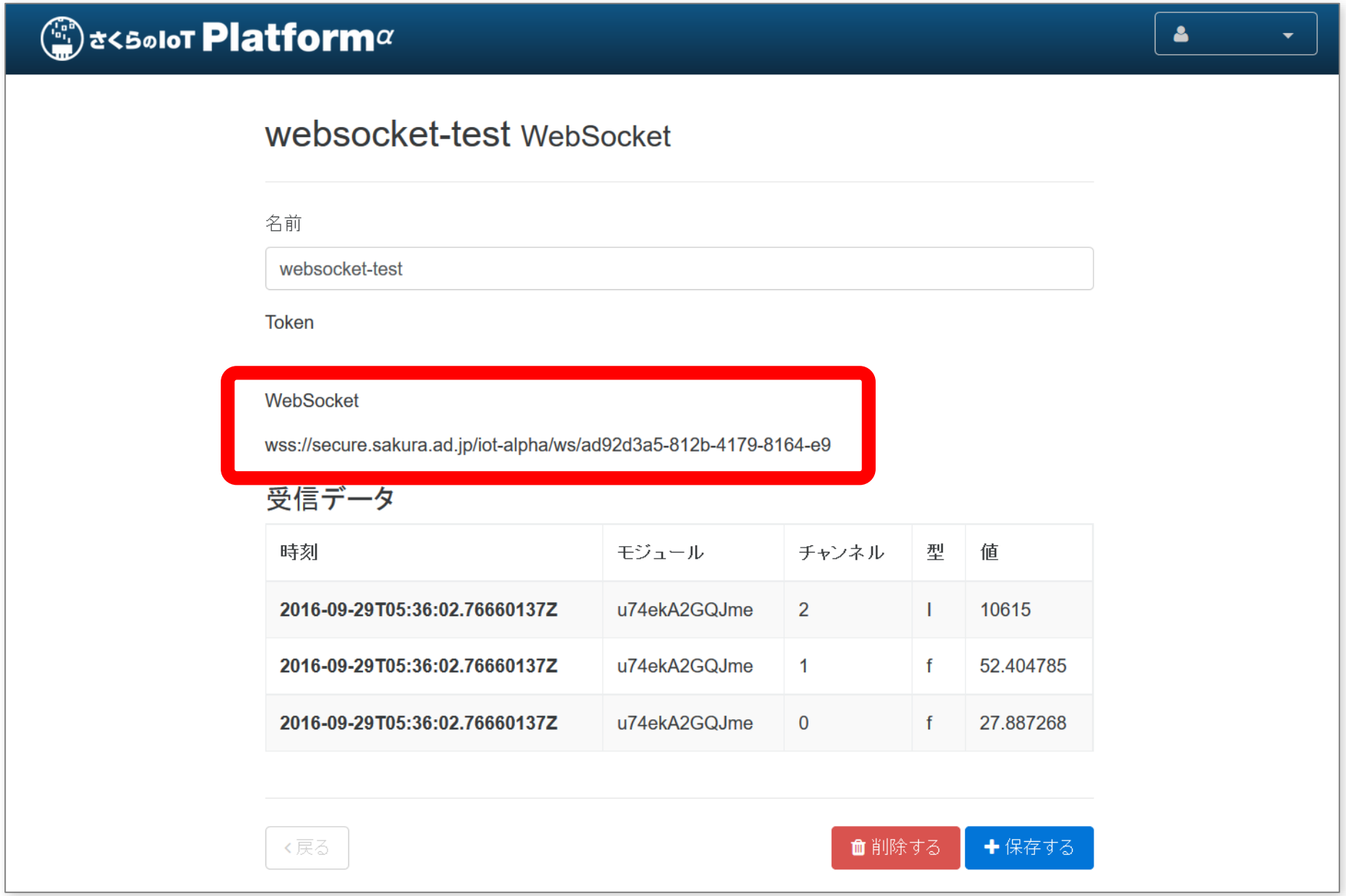
- 田中さん@ZABBIX-JPの記事を参照
  - Zabbix 3.0をCentOS 7にインストール
  - http://qiita.com/atanaka7/items/294a639effdb804cfdaa
- 作業の概略
  - CentOS 7のサーバを作成
  - firewalldの設定
  - Webサーバ(Apacheなど)の入手と設定
  - MariadbのインストールとDB作成
  - Zabbixのインストールと設定

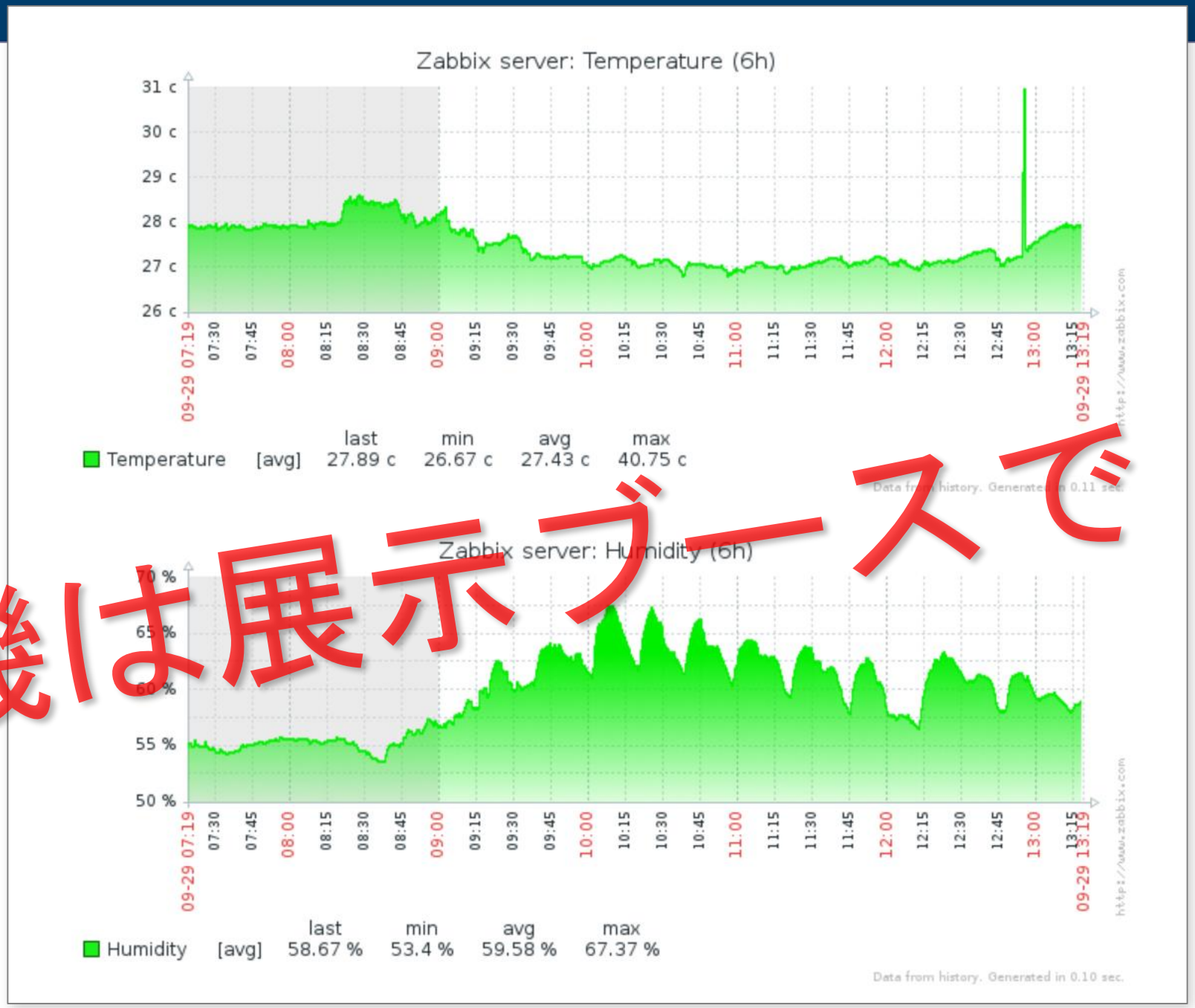
監視の有効化 → アイテムの作成 → グラフの追加 → スクリーンの追加

処理の流れ：
1. さくらの Iot Platform からの WebSocket を受信 (Mojoliciousを使用)
2. 温度と湿度のデータを zabbix_sender で送信

```
#!/usr/bin/perl

use strict;
use warnings;
use Mojo::UserAgent;

my $ua = Mojo::UserAgent->new;
$ua->websocket('wss://secure.sakura.ad.jp/iot-alpha/ws/xxxxxxxxxx/' => sub {
 (略)
                open(CMD, "zabbix_sender -z 127.0.0.1 -s ¥"Zabbix server¥" -k
sakura_iot_temp -o $dat |");
                print "Temp:",$dat,"¥n";
                print "zabbix_sender -z 127.0.0.1 -s ¥"Zabbix server¥" -k sakura_iot_temp -o
",$dat,"¥n";
```

WebsocketのURIを入手しプログラムに設定




Zabbix server: Temperature (6h)
31 c
30 c
29 c
28 c
27 c
26 c
07:30 07:45 08:00 08:15 08:30 08:45 09:00 09:15 09:30 09:45 10:00 10:15 10:30 10:45 11:00 11:15 11:30 11:45 12:00 12:15 12:30 12:45 13:00 13:15
09-29 07:19
09-29 13:19
last min avg max
Temperature [avg] 27.89 c 26.67 c 27.43 c 40.75 c
Zabbix server: Humidity (6h)
70 %
65 %
60 %
55 %
50 %
07:30 07:45 08:00 08:15 08:30 08:45 09:00 09:15 09:30 09:45 10:00 10:15 10:30 10:45 11:00 11:15 11:30 11:45 12:00 12:15 12:30 12:45 13:00 13:15
09-29 07:19
09-29 13:19
last min avg max
Humidity [avg] 58.67 % 53.4 % 59.58 % 67.37 %
Data from history. Generated in 0.10 sec.

実機は展示ブースで！！

まとめ

- さくらのIoT Platformの概要
  - 開発経緯
  - 主な機能/システム構成/パートナー連携
  - 事例
  - β版の販売について
- 実際に使ってみる
  - マイコンおよびプログラムの構築
  - さくらのIoT Platformの設定
  - Webサービスとの連携

さくらのIoT Platformが目指す世界

既存の事業領域/スキルセットの大幅な変更なく
モノ/サービスづくり、連携に注力可能

マイコンおよび
プログラムの構築

温湿度センサ
（HDC1000）

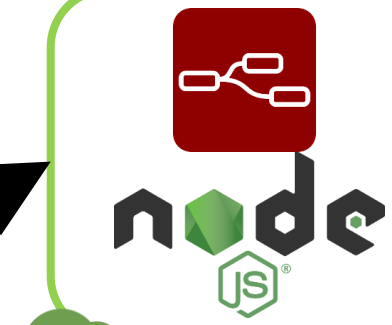
マイコン
（Arduino Uno）

さくらの通信
モジュール

さくらのIoT
Platformの設定

さくらのIoT
Platformβ

Webサービス連携
（Node-RED）



仮想サーバ

ZABBIX

仮想サーバ

Webサービス連携
（Zabbix）

「さくらのクラウド」2万円クーポンも配布中！
ステッカーやグッズも配布中！

- 広島でもさくらのイベントを開催したい！
  - さくらのIoT Platformのハンズオン
  - さくらのクラウドやArukasなどのハンズオン
  - さくらのタベ / さくらクラブ など…
- 協力者求む！
  - 会場の提供
  - 参加者集め
  - 地元コミュニティとの共催も可

そこに、さくら