さくらのIoTプラットフォーム「sakura.io」を使ってみよう

https://sakura.io

DAY	COMPANY	DEPARTMENT	NAME
2018/12/08	さくらインターネット株式会社	コミュニティマネージャー	法林 浩之

本日の資料はこちらで公開します

https://www.slideshare.net/hourin/

もしくは

「slideshare 法林」で検索

#sakuraio　#osc18fk

法林 浩之

@hourin

どんな人？

・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
  – 会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)
  – さくらのナレッジ 編集長
・日本UNIXユーザ会 幹事 (元会長)
  – さまざまなコミュニティと共同でイベントを開催
  – 全国各地のイベントで研究会を開催
・くわしくは「法林浩之」で検索

- IoTを取り巻く状況
- IoTサービスを作るときの問題点
- sakura.ioについて
  - 開発経緯
  - サービス概要
  - 利用事例
- sakura.ioとOSSの組み合わせ

SAKURA internet

大阪本社
（梅田/大阪）

東京支社（西新宿）

福岡オフィス（赤坂）

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	22億5,692万円
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並びに製作・販売
従業員数	495名（連結/2017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）
グループ会社	アイティーエム株式会社 https://www.itmanage.co.jp/ 株式会社S2i http://www.s2i.life/ ゲヒルン株式会社 https://www.gehirn.co.jp/ 株式会社Joe'sクラウドコンピューティング https://joes.co.jp/ ビットスター株式会社 https://bitstar.jp/ 櫻花移動電信有限公司

さくらインターネットの事業領域

SAKURA internet

レンタルサーバ

さくらのレンタルサーバ
さくらのマネージドサーバ

VPS・クラウド（仮想化基盤）

さくらのVPS

さくらのクラウド sakura cloud

専用サーバ・データセンター

さくらの専用サーバ sakura dedicated server

ハウジング
リモートハウジング

新規サービス

IoT sakura.io

通信モジュールから提供することで、セキュアな通信環境とデータの保存や処理システムを一体型で提供するIoTプラットフォームサービス

IoT さくらのセキュアモバイルコネクト

お使いのデバイスを、SIMからさくらのクラウドのまでインターネットを経由せず、安全に接続できるIoT向けモバイルサービス

AI・人工知能 高火力 KOUKARYOKU

機械学習、データ解析、高精度シミュレーション用途に特化したGPU搭載の専用サーバサービス

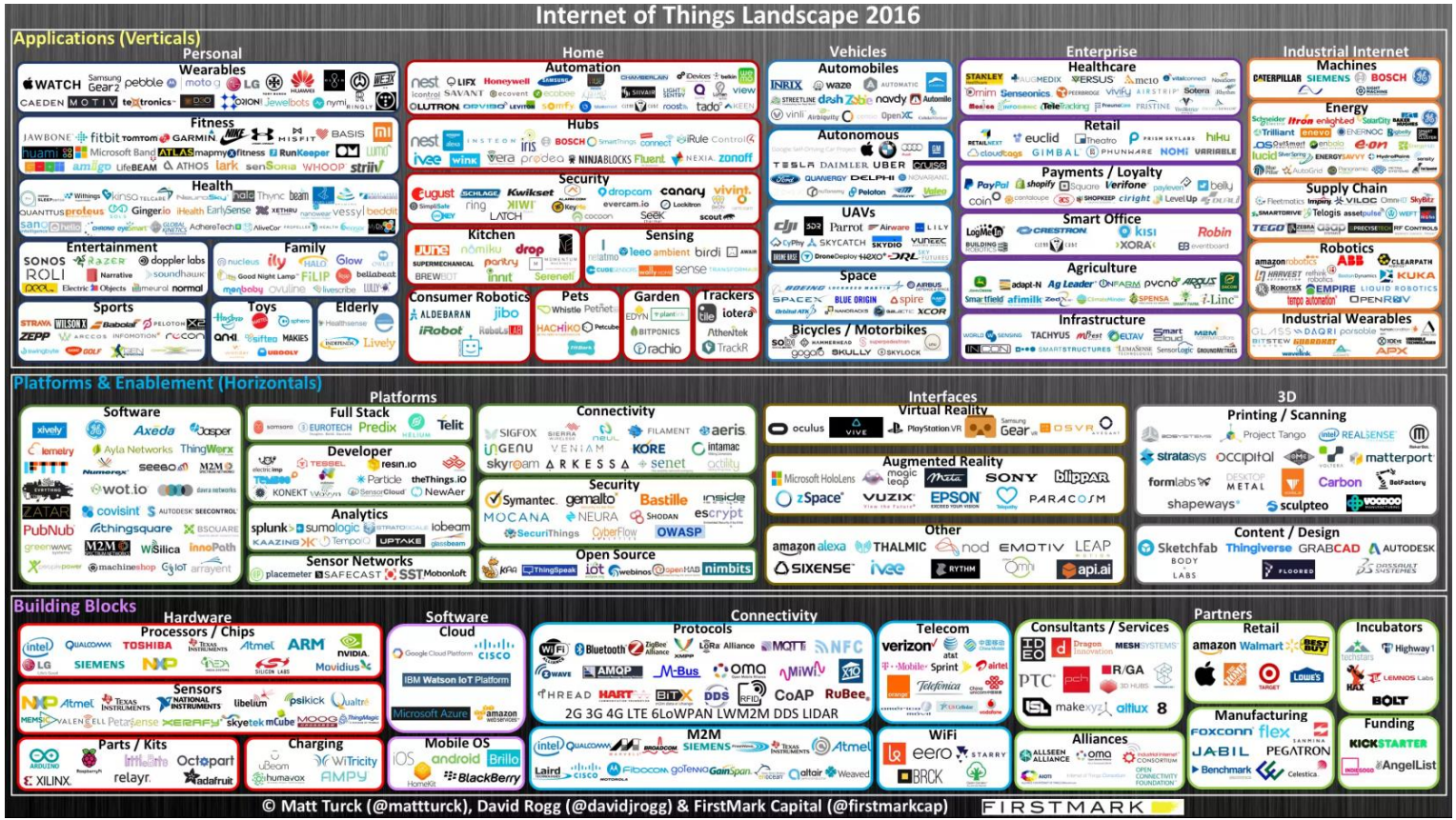
コンテナ Arukas HOSTING DOCKER CONTAINERS

Dockerコンテナのマネージドサービスであり、コンテナを手軽かつシンプルに実行・運用

新しい社会のインフラを支えながら、最先端のサービスを構築

IoTを取り巻く状況

IoTと呼ばれているもの



http://mattturck.com/2016-iot-landscape/

“インターネット”と同じぐらい広すぎる

IoTと呼ばれているもの(垂直統合型)

モノとサービスの両方を提供
(垂直統合型)

IoTと呼ばれているもの(水平統合型)

IoTと呼ばれているもの(要素技術)

sakura.ioの位置付け

http://mattturck.com/2016-iot-landscape/

SAKURA internet

IoTの領域 → 収集

モノゴトをデバイス経由で
データ化し収集、蓄積

AIの領域 → 情報化

データのクレンジング処理
他情報と組み合わせによる意味付け

情報

ロボティクスの領域 → フィードバック

情報を活用し具体的な
アクションに反映

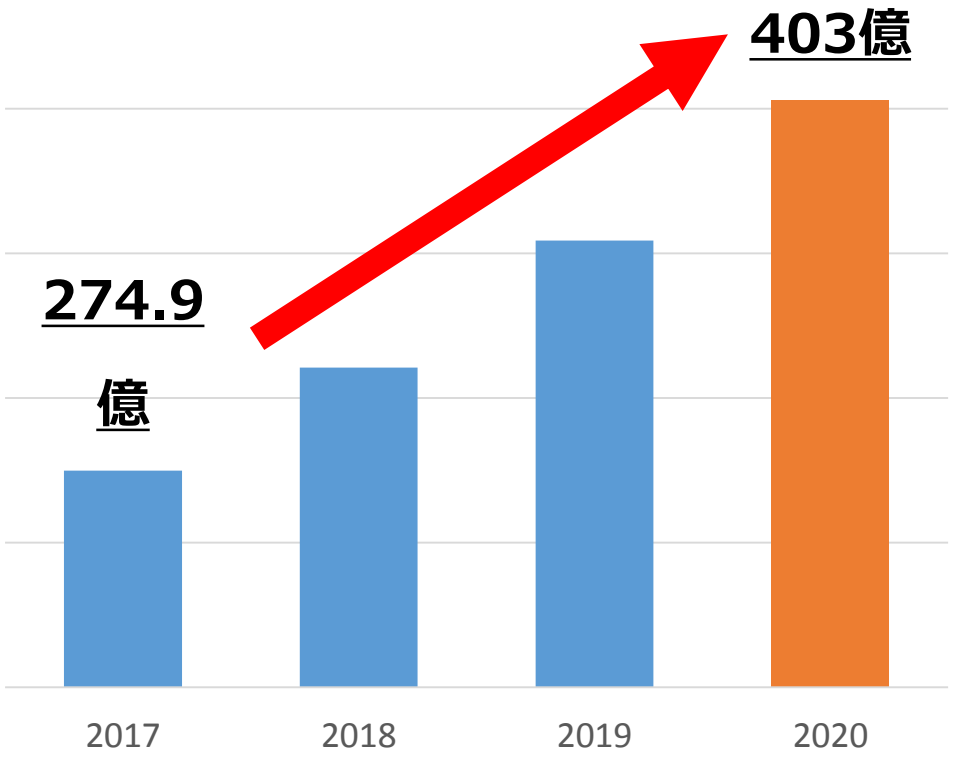
生活

and,more..

年平均成長率予測

13.6%

世界のIoTデバイス数の推移及び予測



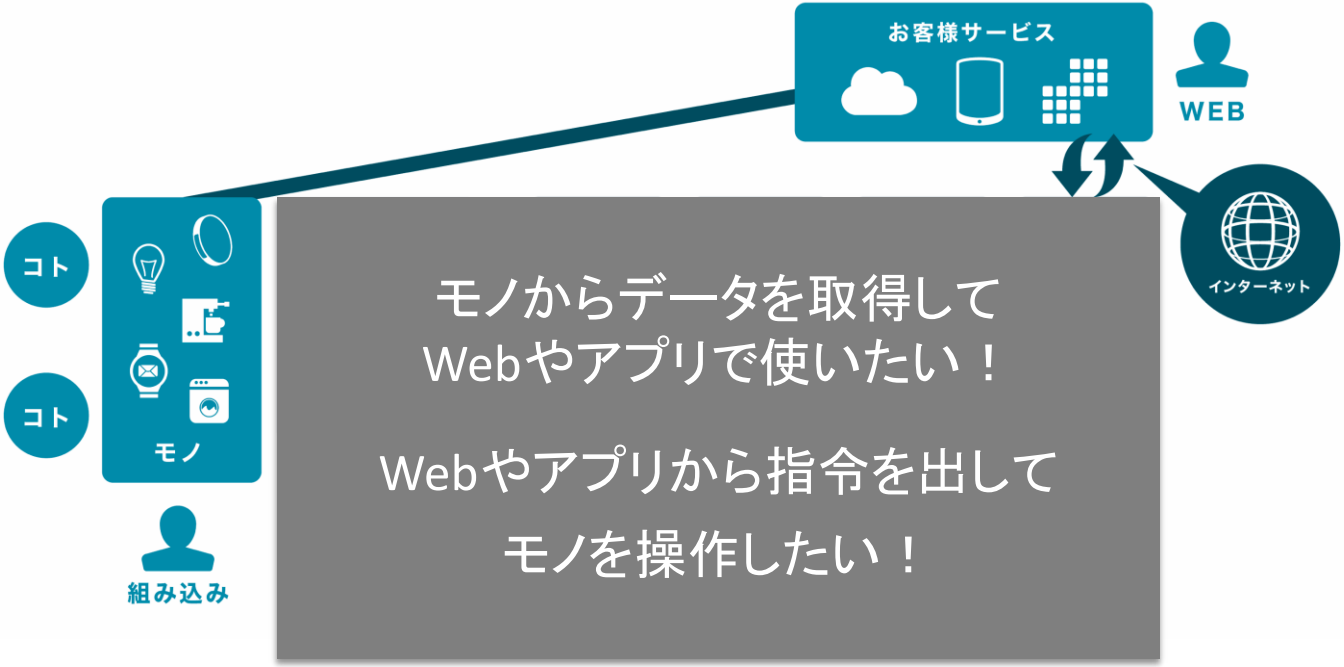
出典：「平成30年版情報通信白書」（総務省）

どうやって
データ
集めるの？

http://mattturck.com/2016-iot-landscape/


IoTサービスを
作るときの問題点

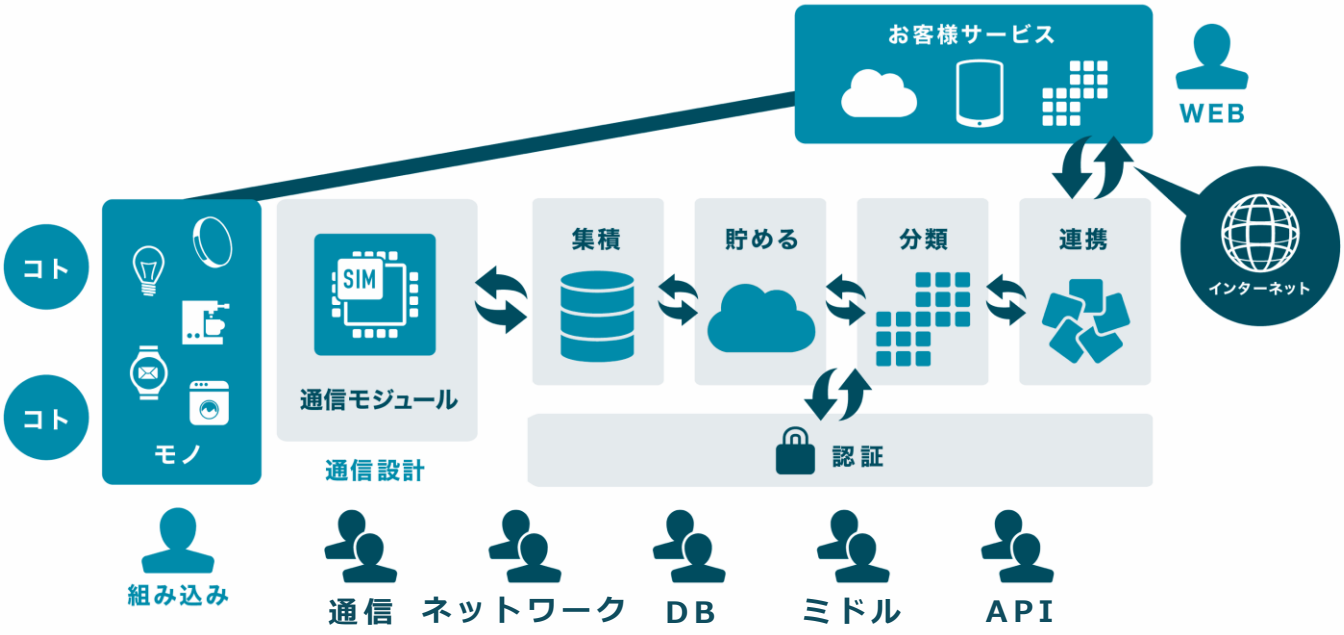
IoTサービスを作る人がやりたいこと



モノをインターネットにつないで何かしたい

IoTサービスを作る人が実際にやること

SAKURA internet



ハードウェア/ソフトウェア両方の知識が必要

IoTサービスを作るときに必要な作業

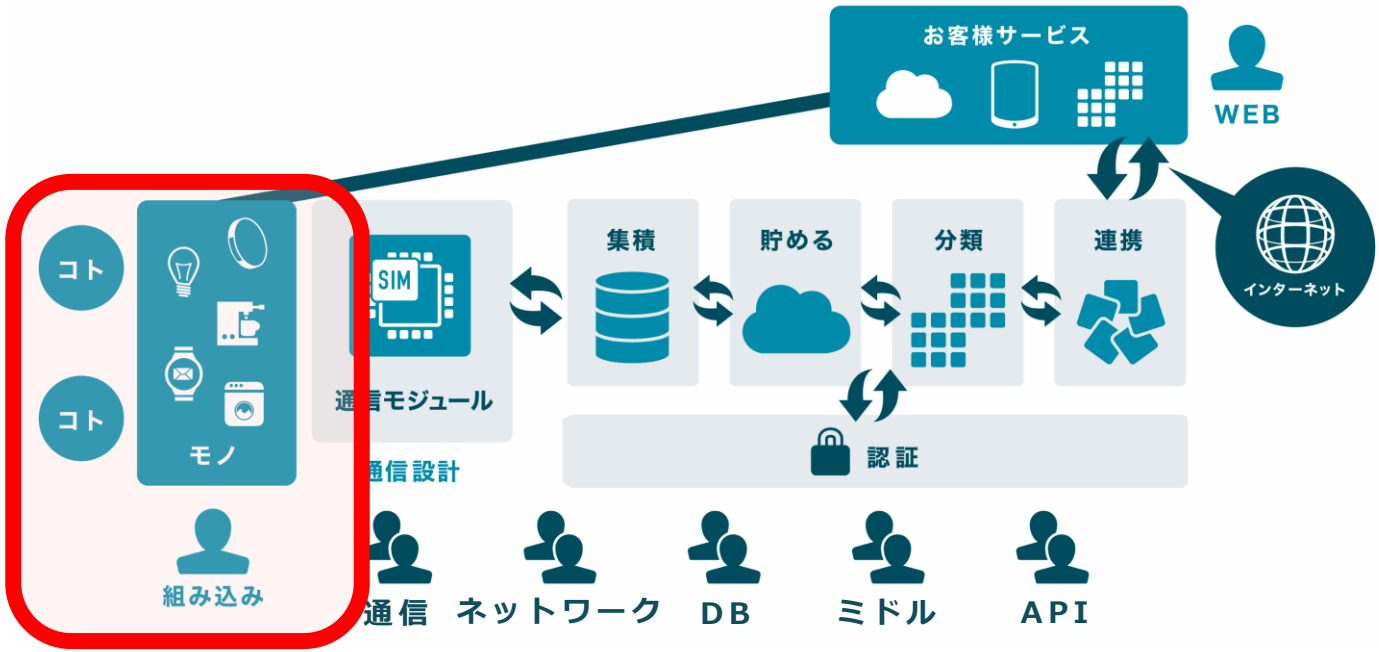
- ハードウェア(モノ作り)
  - 機構設計 / 電気設計 / ファームウェア設計 / 通信設計
- ソフトウェア(フロントエンド)
  - UI設計 / アプリ開発
- ソフトウェア(バックエンド)
  - インフラ設計 / ミドルウェア設計 / DB設計 / API設計
- セキュリティ
  - ユーザ認証 / 機器認証 / 暗号化 / バックアップ

IoTサービスを作るときに必要な作業

- ハードウェア(モノ作り)
- セキュリティ
  - ユーザ認証 / 機器認証 / 暗号化 / バックアップ

これらを全部できる人は
ほとんどいない

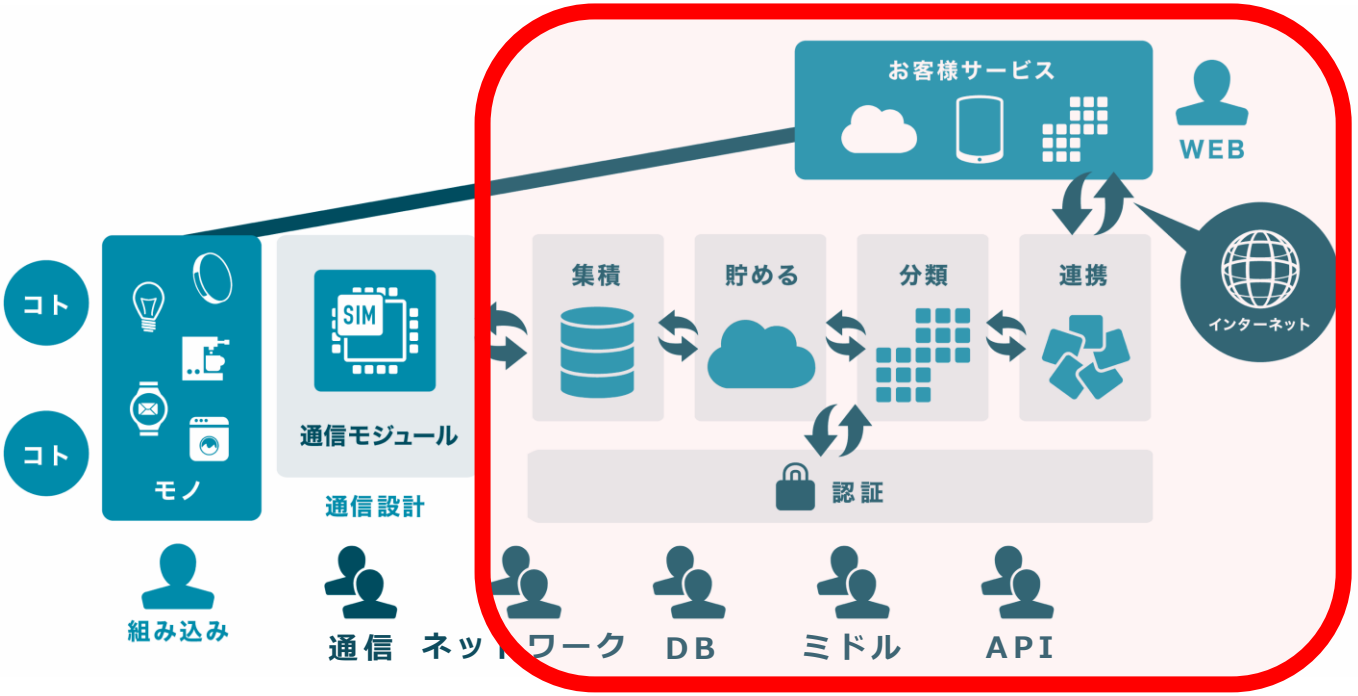
ハードウェアエンジニアの守備範囲



物理的なことや電気的なことはわかるが
TCP/IPやHTTPのことはわからない

ソフトウェアエンジニアの守備範囲

SAKURA internet



TCP/IPやHTTPのことはわかるが
物理的なことや電気的なことはわからない

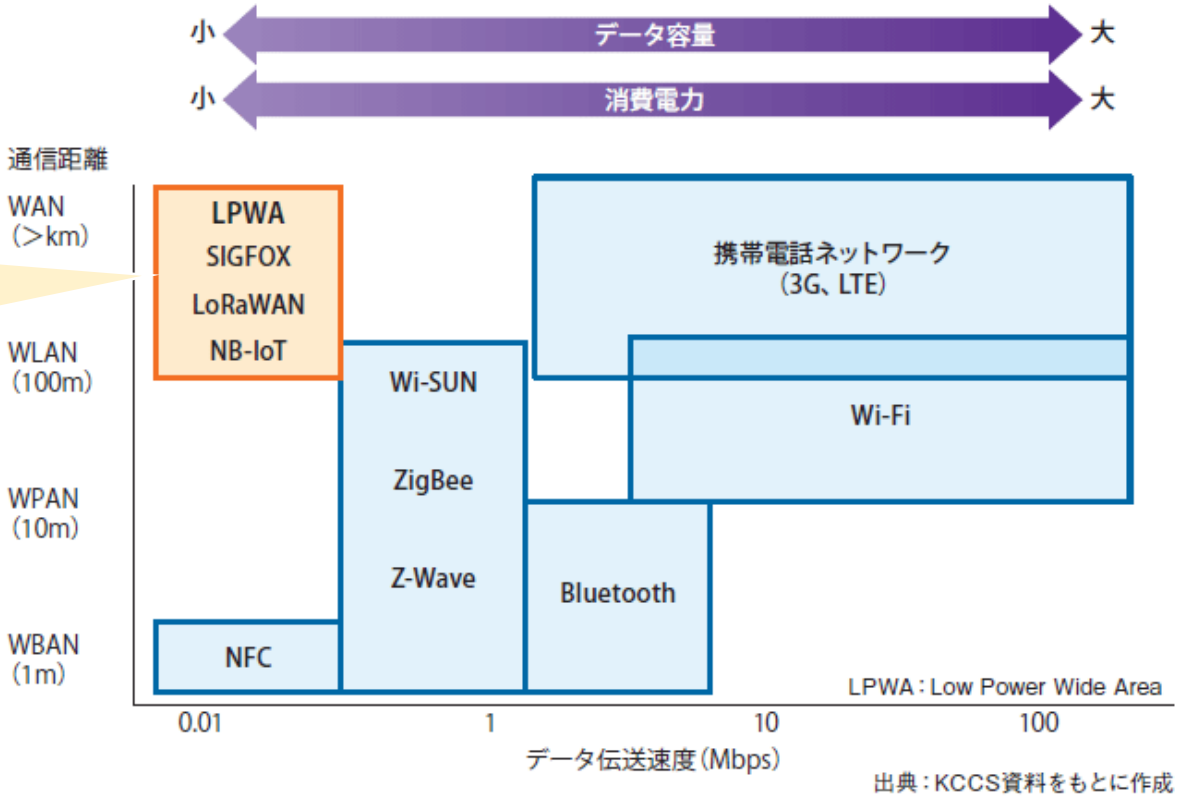
モノをネットにつなぐ手段は増えている

	TCP/IP	not IP
近くとつながる	 Wi Fi™	LoRa™ ZigBee®
どこでもつながる	4G LTE	LoRaWAN™ sigfox

通信方法をどうするか

SAKURA internet

IoT向けの通信方式として注目



出典：ビジネスネットワーク.jp　http://businessnetwork.jp/Detail/tabid/65/artid/5106/Default.aspx

SAKURA internet

どの方式にも弱点がある

- LAN / WiFi：回線やアクセスポイントが必要
  - モノをつなぐためだけにBフレッツを敷設？
- LTE
  - 山奥や地中深くでは電波が入らない
  - 空中では使用禁止(電波法)
- Bluetooth：近距離でないと使えない
- その他：普及していない

- IoTで扱うデータは個人情報が多い
  - 生体情報、位置情報、など
  - データの暗号化は必須
- IoTデバイスはコンピュータとしては非力な場合が多い
  - Linuxでは重いので軽量なリアルタイムOSが普及
  - デバイス側に暗号化の仕組みを実装するのは難しい
- ハードウェアエンジニアにはTCP/IPやSSLの実装は困難
- インターネットにつなぐと攻撃対象になる
  - 不正アクセスを受けて乗っ取られる
  - DoS攻撃を受けて通信できなくなる

- 多様な環境に適応する必要性
  - WiFiがあるとは限らない
  - スマホがあるとは限らない
- できれば設定や操作をなくしたい
  - デバイスが勝手にデータを送受信すればよい
- 初期投資が大きい
  - モノの製造コストがかかるのが大きな負担
- サービスの継続にコストがかかる
  - デバイスの量産、保守、サポートなど

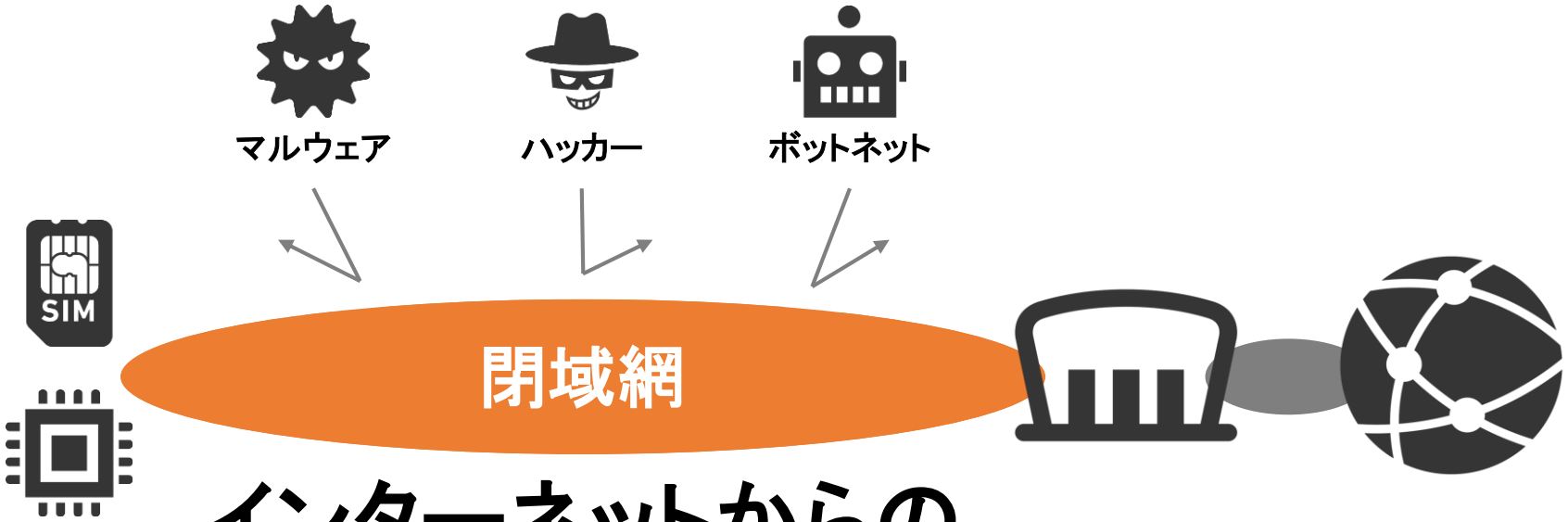

さくらインターネットが考える
IoTソリューション

1. どこでも、だれでも
2. 安全なネットワーク
3. ノーコミット＆ノーリミット
4. 検証から、量産まで
5. 圧倒的コスパ

LTEのメリット

1. 高いカバー率（島嶼、山間）
2. 移動体でも使用可能
3. 設定不要で繋がる

LTEなら最終利用者も手軽に利用できる



インターネットからの
直接アクセス不可

デバイス⇔クラウド間のリスクポイントを削減

1. 契約期間縛りなし
2. 契約/解約手数料なし
3. 再登録が何度でも可能
4. 速度制限なし

IoT用途での課題となる各種制約や上限を撤廃

さくらのクラウド セキュアモバイルコネクト

IoT向けSIMの提供

デバイスからコンピューティングリソースにセキュアに接続できるモバイル回線

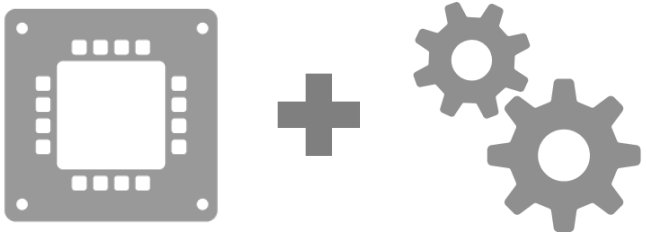
カード型SIM

チップ型SIM

sakura.io

プラットフォームの提供

モジュール＋クラウドの一体提供で開発スピードを向上、スケール対応や運用保守はさくらで実施



専用モジュール
（単体方式 or ゲートウェイ方式）

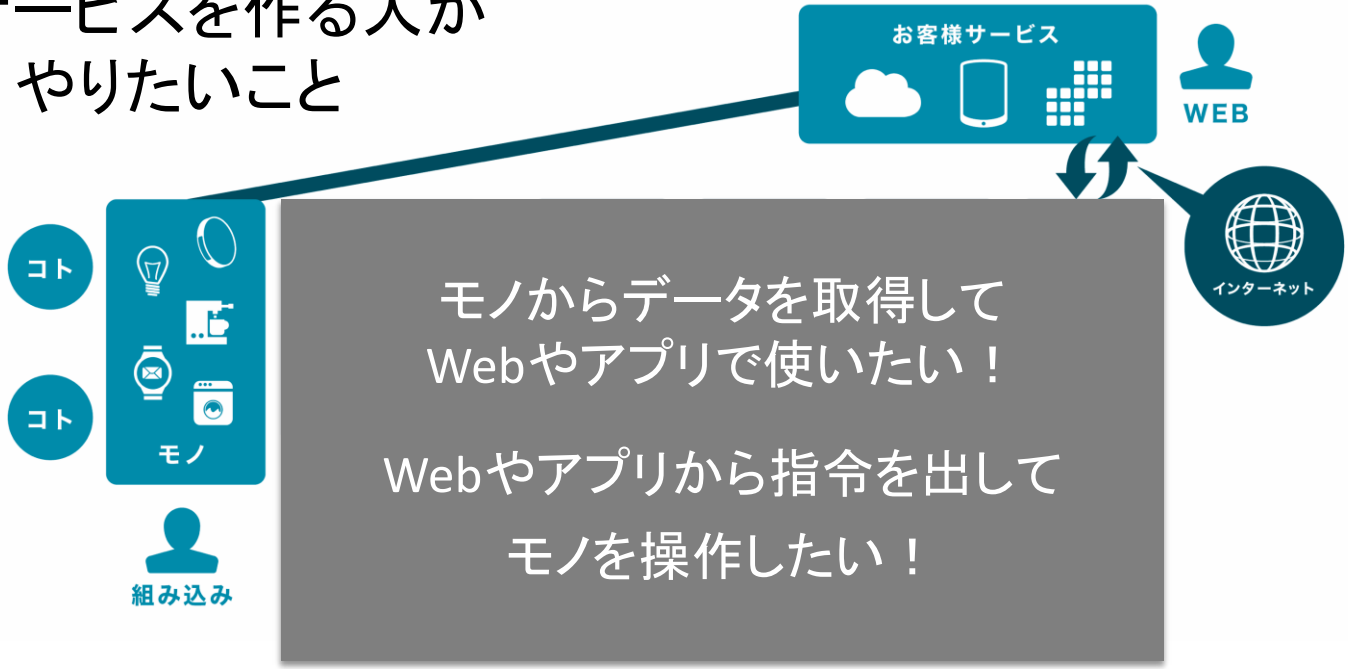
基本機能
（つなぐ・貯める・連携する）


sakura.io
（さくらアイオー）

IoTサービスを作るときの問題点

SAKURA internet

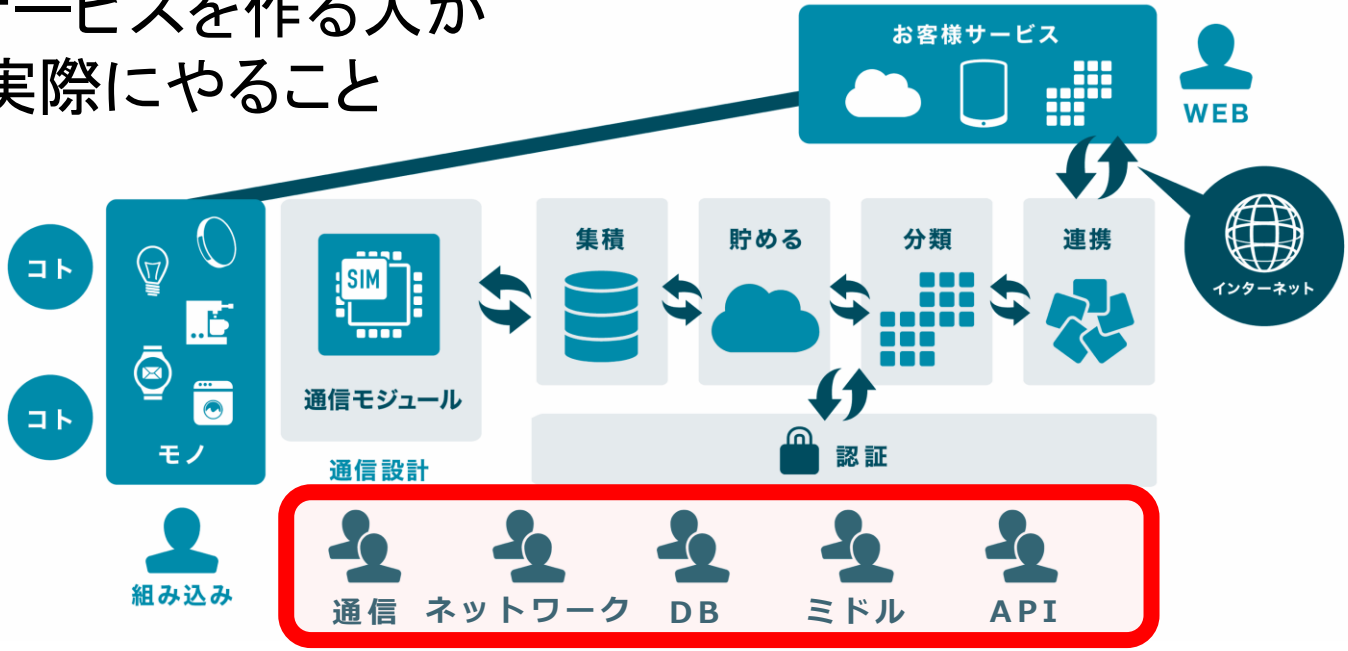
IoTサービスを作る人が やりたいこと



モノに関する部分と、Webサービスやアプリの部分だけ作りたい

SAKURA internet

IoTサービスを作る人が実際にやること

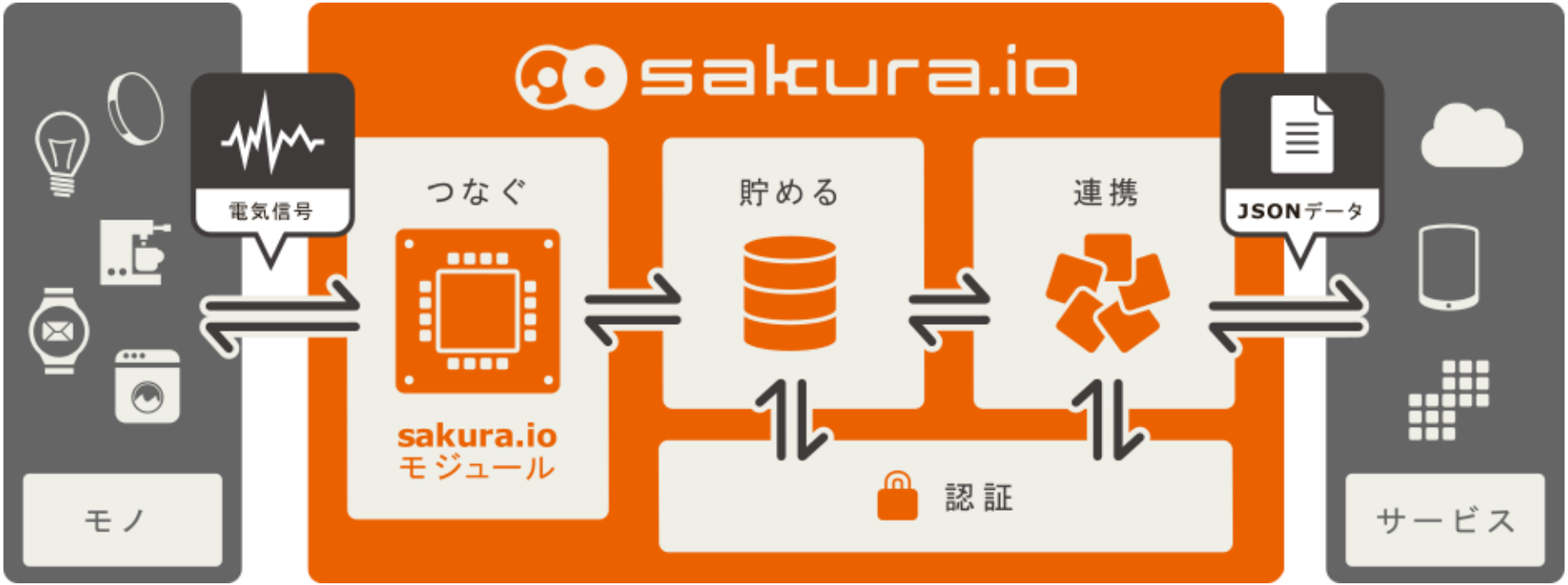


ネットワークとデータをやり取りしたいだけなのに…
やらなければならないことが多すぎる

さくらインターネットの取り組み

モノとWebの間でデータを相互にやりとりするための
プラットフォームサービスを開発

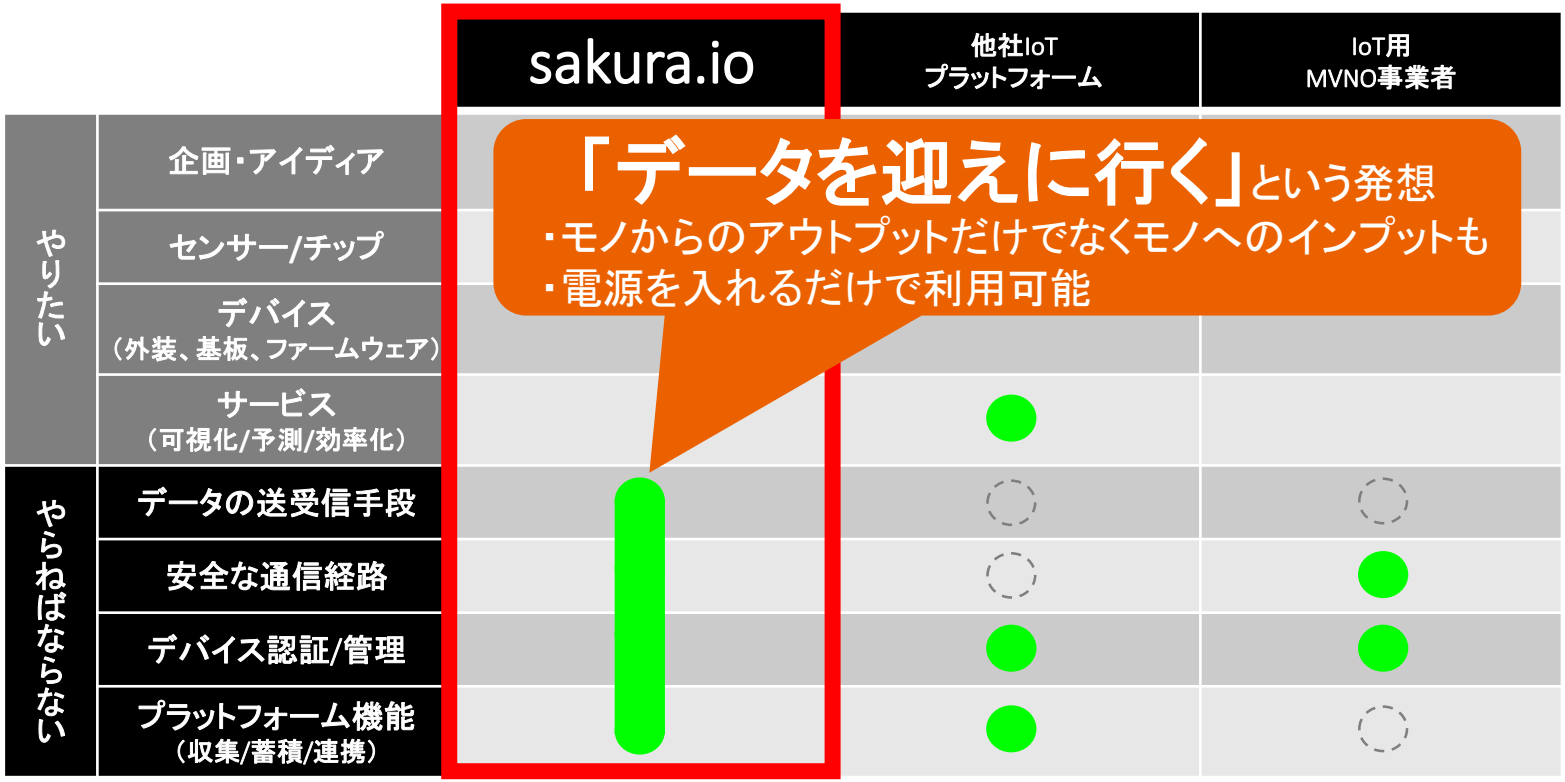
sakura.ioが提供する価値



組み込み側はモノとの**電気信号** / Web側はサービスとの**JSONデータ**
のやり取りに注力可能

他のIoTプラットフォームとの差異

SAKURA internet

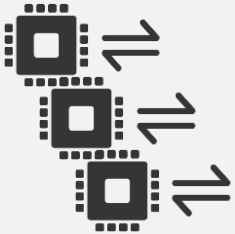


【 やりたい 】に注力できるプラットフォームとして提供

sakura.io
サービス詳細

データセンタ事業者だからできた

SAKURA internet

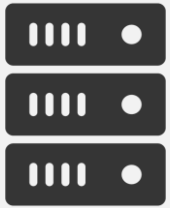


データの収集

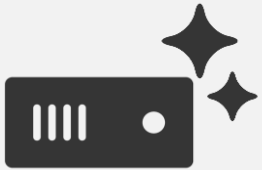
データの蓄積

データの連携

運用機能



ラージスケール対応



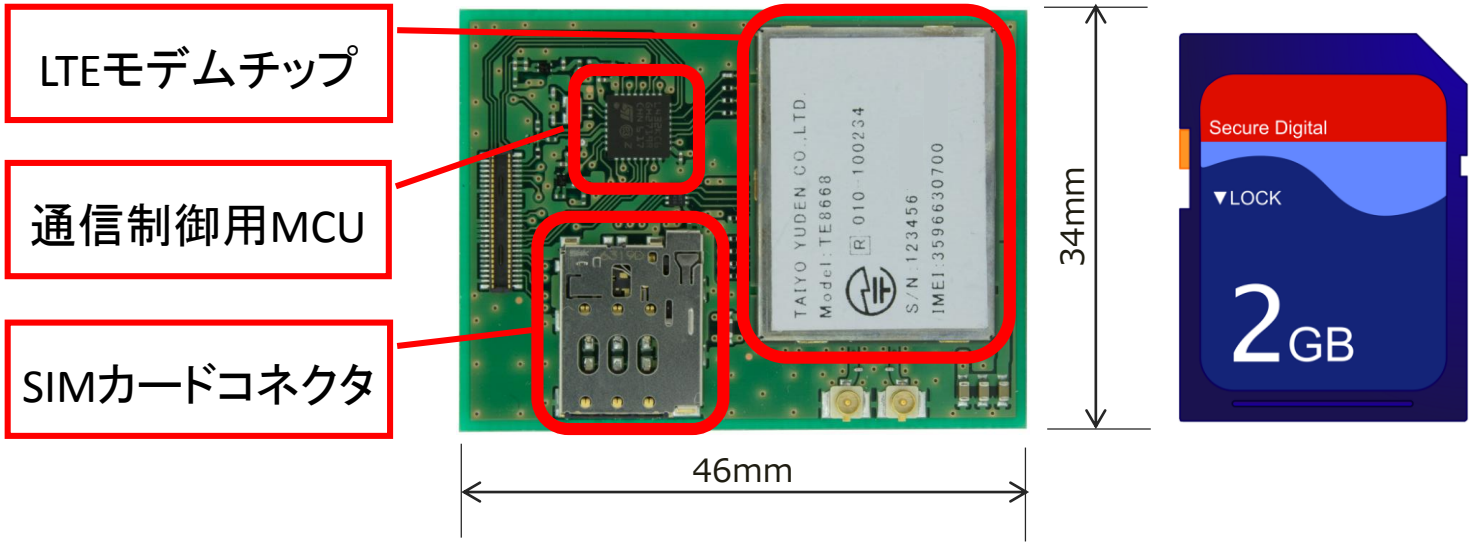
アップデート

障害切り分け、復旧

セキュリティ

必要な機能・運用をプラットフォームサービスとして提供
IoTデバイスやサービスごとの基本機能開発や運用設計は不要

データの収集 | データの蓄積 | データの連携 | ラージスケール対応 | アップデート | 障害切り分け、復旧 | セキュリティ | 運用機能



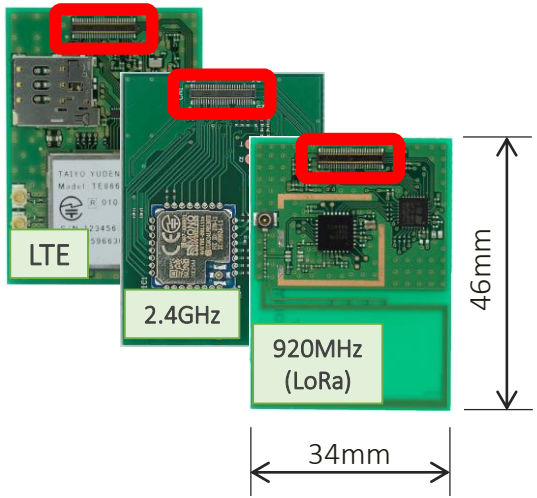
SDカードほぼ2枚分に収まるコンパクトサイズ
モノ側の通信に必要なすべてを凝縮

どこでもつながる通信モジュール

SAKURA internet

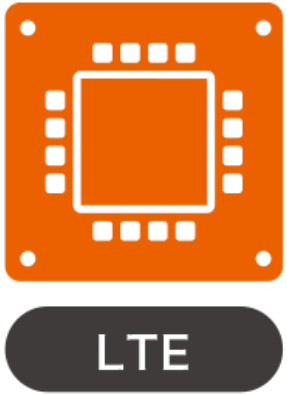
データの収集 | データの蓄積 | データの連携 | ラージスケール対応 | アップデート | 障害切り分け、復旧 | セキュリティ | 運用機能

量産性に配慮した
“**基板間コネクタ**”を採用

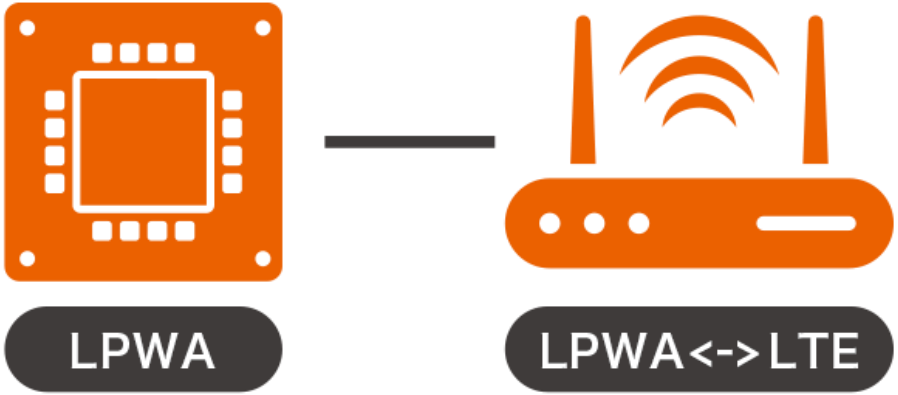


方式	GW	特徴	通信可能レンジ	伝送速度	消費電力
LTE	不要	単独使用可	キャリア網内どこでも	速い	大きい
2.4GHz帯	必要	短距離大容量	数百メートル（最大1Km程度）	速い	小さい
920MHz帯（LoRa）	必要	長距離小容量	数キロメートル（最大10km程度）	遅い	小さい

共通インターフェースおよび寸法のため
複数の無線規格への対応が容易



単体方式（LTE）

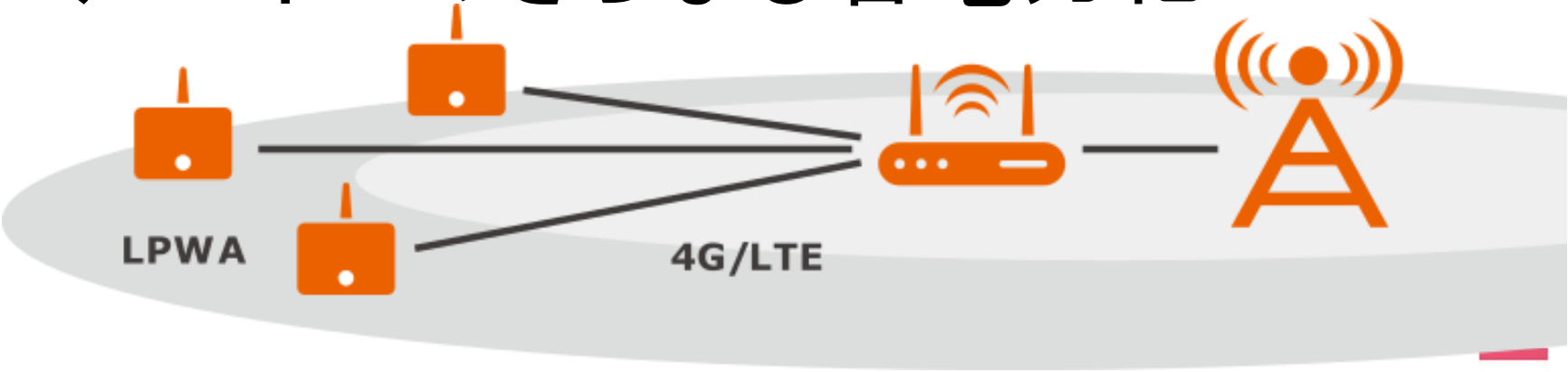


ゲートウェイ方式（LTE＋LPWA）

デバイスは**単体方式**と**ゲートウェイ方式**を提供

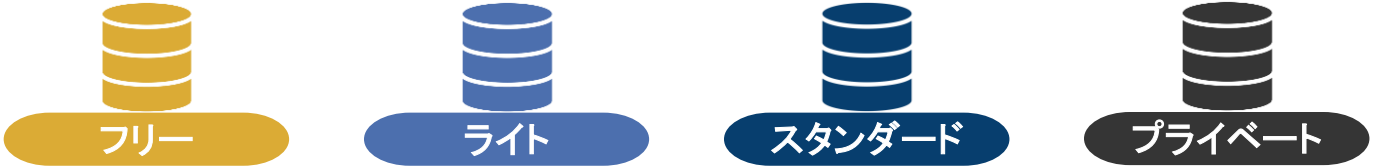
LPWA併用のメリット

1. ラストワンマイル対策
2. デバイスのさらなる省電力化



データはポリシーに応じたデータストアに自動で保管

SAKURA internet



	フリー	ライト	スタンダード	プライベート
料金	無料	有料（50円/月）※	有料（200円/月）	有料
専有/共有	共有領域	共有領域	共有領域	専有領域
公開有無	公開	非公開	非公開	非公開
閲覧可能期間	40日間	40日間	2年間	制限なし
リリース	未公開	リリース済み	リリース済み	未公開

※料金は通信モジュール1個あたりの金額となります
※ライトプランは現在無料で提供されています

デバイスから送られたデータはポリシーに応じて
プラットフォーム内のデータストアに自動で保存される

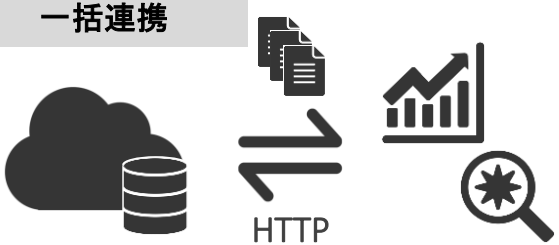
Webエンジニアの負担を軽減する設計

SAKURA internet

データの収集 | データの蓄積 | データの連携 | ラージスケール対応 | アップデート | 障害切り分け、復旧 | セキュリティ | 運用機能

リアルタイム連携

一括連携



JSON形式

```
{
   "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,
             ...
         }]
   }
}
```

データ取り出しやデバイス制御はすべてJSONフォーマットで実施
既存システムや扱いに慣れたクラウドサービスとの接続も可能

「安全」を超えた「安心」を

SAKURA internet

データの収集 | データの蓄積 | データの連携 | ラージスケール対応 | アップデート | 障害切り分け、復旧 | セキュリティ | 運用機能

デバイスからプラットフォームまでは閉域網経由、他デバイス含め
外部アクセスはAPI経由でのみ行うことでセキュリティを担保

- インターネット上のサーバ間の通信はSSLを使用
  - サーバは継続的にアップデート可能

- 組み込み機器との通信は構造上安全にする
  - 閉域網を利用(グローバルネットワークに接続しない)
  - 通信の暗号化と認証は基本的にLTEを利用
  - その上で簡易な暗号化と認証をソフトウエアで実装
  - モノ作りの人にTCP/IPやSSLを実装させない

運用を考慮した機能群

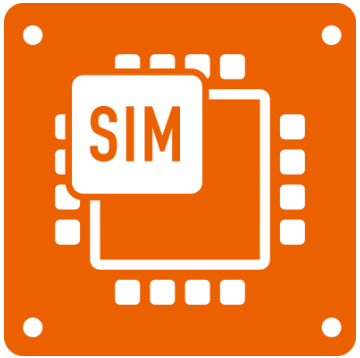
SAKURA internet

データの収集 | データの蓄積 | データの連携 | ラージスケール対応 | アップデート | 障害切り分け、復旧 | セキュリティ | 運用機能

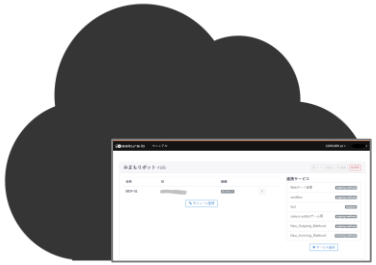
時刻提供機能

時刻要求

時刻提供



時刻提供



※一部マイコン側での対応が必要です

スケジュールでの一斉動作といった場合でも
通信モジュールから時刻情報を提供可能

運用を考慮した機能群

SAKURA internet

データの収集 | データの蓄積 | データの連携 | ラージスケール対応 | アップデート | 障害切り分け、復旧 | セキュリティ | 運用機能

簡易位置情報提供機能

※利用には別途月額料金が必要になります

エリアレベルでの大まかな設置場所を収集するような
ケースであれば通信モジュールの機能として提供

運用を考慮した機能群

SAKURA internet

データの収集 | データの蓄積 | データの連携 | ラージスケール対応 | アップデート | 障害切り分け、復旧 | セキュリティ | 運用機能

ファイル配信機能

※一部マイコン側での対応が必要です
※別途通信量に対する課金が発生します

設置済みデバイスに対する遠隔アップデート等
ソフトウェアな問題への対処を実現

世界のすべてを繋ぎたい

SAKURA internet

汎用/特定サービスとの連携

Microsoft Azure

myThings

IBM Bluemix

powered by amazon web services

[New!] Google Cloud Platform

WebSocket

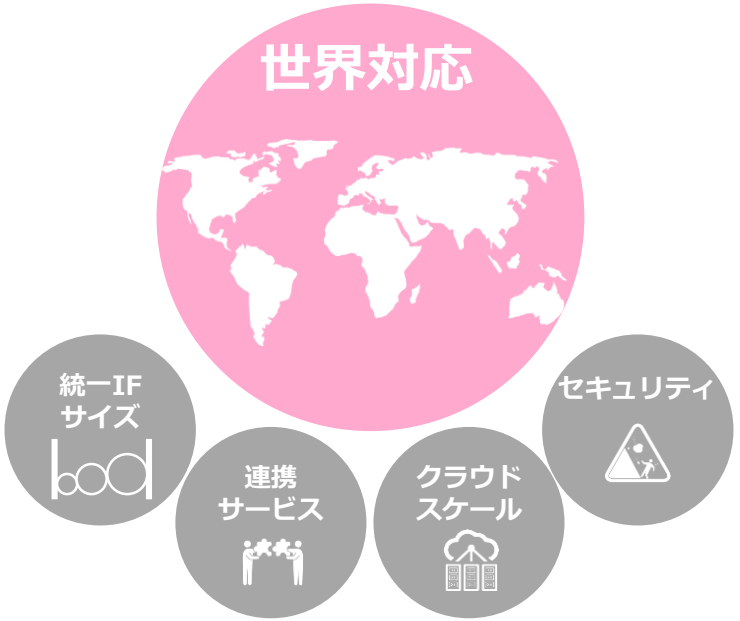
MQTT.ORG

webhooks

Node-RED

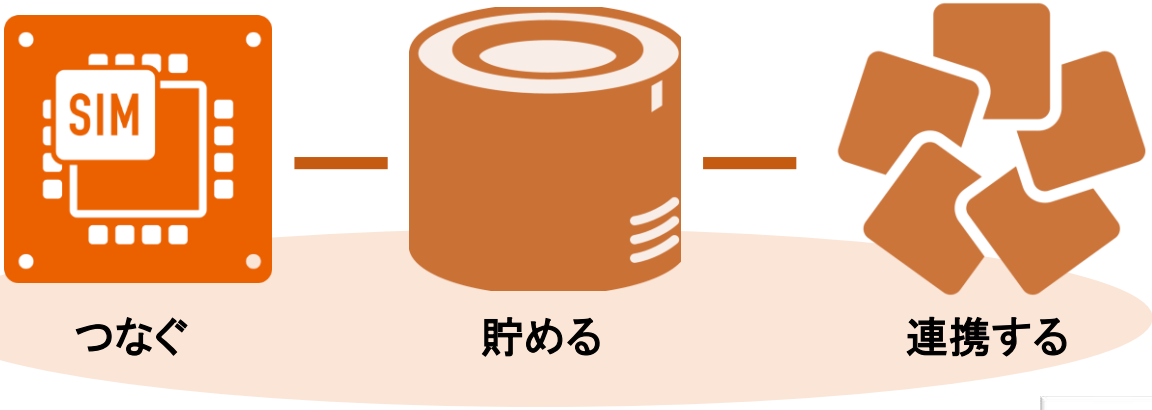
＋

世界中で利用できる



既存システムと連携しやすいフラットな仕組みを
世界のどこでも使えるように提供

ご提供価格/方式



初期費用（モジュール購入） 8,000円

月額費用（回線、プラットフォーム利用） 60円/月 ～

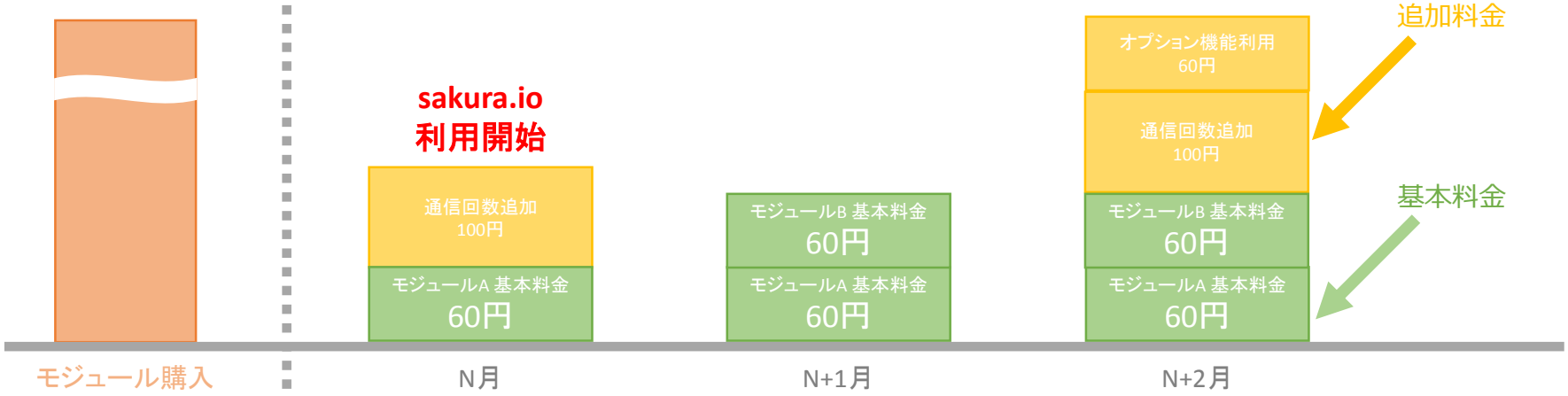
※料金はモジュール単位

sakura.io 料金体系（月額費用）

SAKURA internet

- **基本料金**
  - ✓ 60円/月　※毎月1万回分の通信が可能なポイントを付与
- **追加料金**
  - ✓ 通信回数の追加、オプション機能の利用

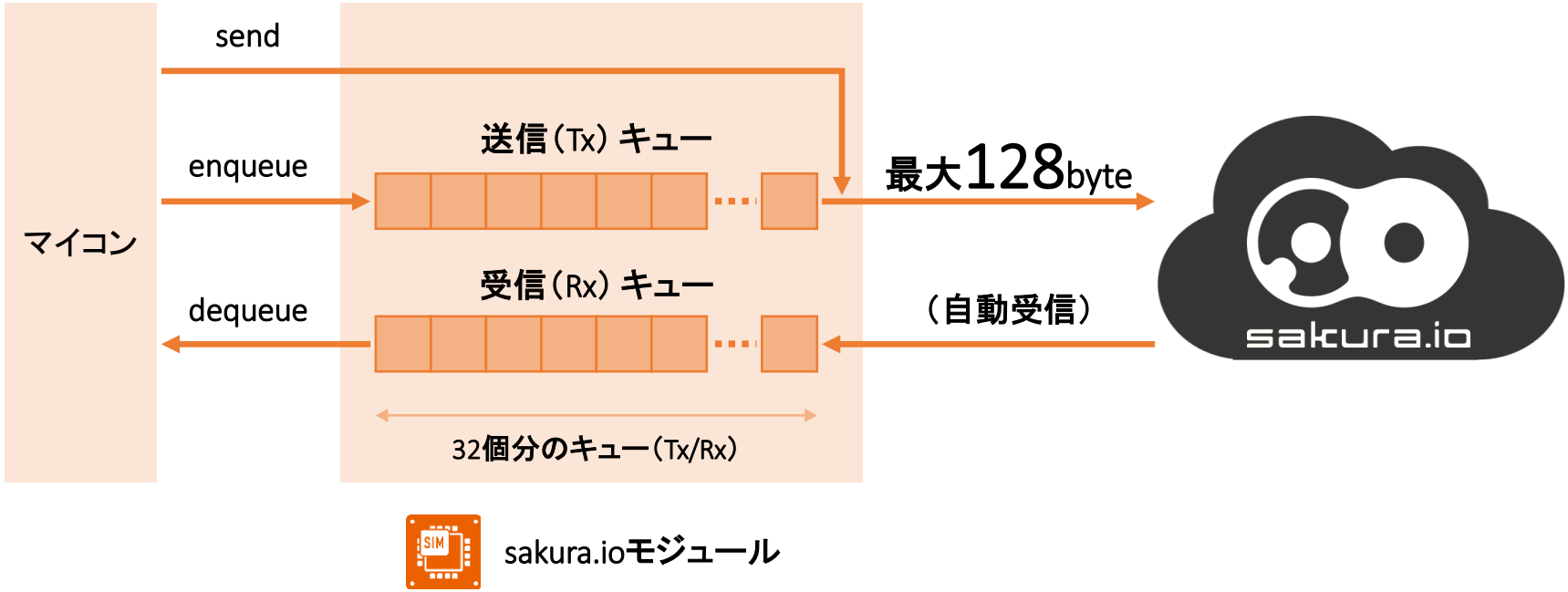
※特定のオプション機能を利用した場合は別途定額の追加料金がかかります



5分に1回の通信なら、毎月60円で実現

より幅広いサービスへ適用可能

sakura.io 1回の通信に含めることができるデータ（LTEモデル）

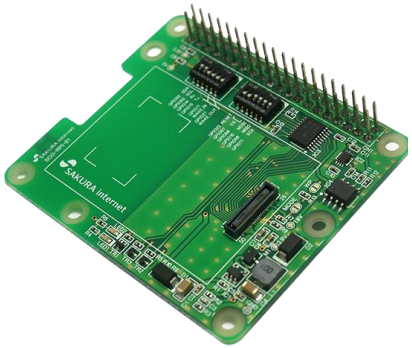


1回の送受信データ（メッセージ）は、最大「128byte（8byte x 16）」
時間や送信元情報はプラットフォーム受信時に付与、マイコン側対応不要

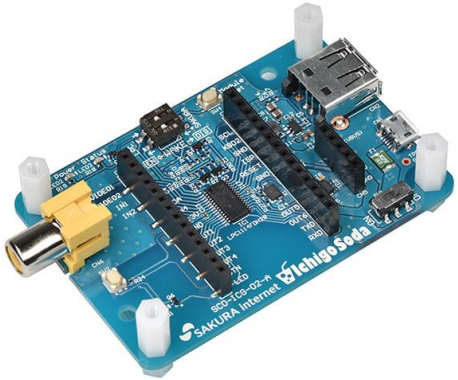
sakura.io ハードウェア料金表

LTE通信モジュール

Arduino用



Raspberry Pi用



IchigoJam用

商品名	必須/オプション	料金
sakura.io モジュール（LTE）	必須	8,000円　登録後 60円/月
sakura.io シールド for Arduino	オプション	5,000円
sakura.io HAT for Raspberry Pi	オプション	5,000円
IchigoSoda / IchigoJam for sakura.io	オプション	5,000円

※モジュール金額は個包装のものです
※すべて税別表記です

SAKURA internet

提供方式	数量	用途
個包装	1個～	少量生産（数台～十数台） 個人利用/プロトタイプ
トレイ	90個単位	少量～中量生産（数百台） サービス提供開始
ライセンス （プロトコルライセンス）	不問	自由な部品/レイアウト 大量生産製造ライン投入

段階に応じて提供方式を選択可能

SAKURA internet

ライセンス方式は通信機能の作り方を提供
「自社ソフトウェアに組み入れたい」に対応

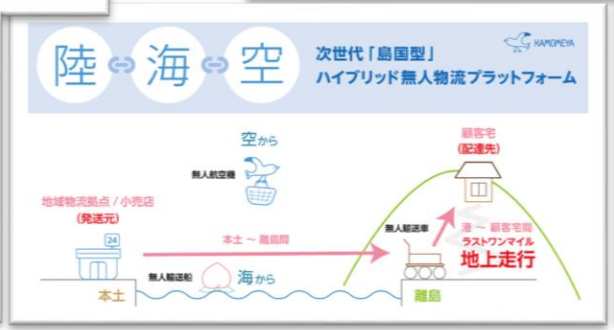
	さくらインターネット	お客さま
開発時	● プロトコル仕様書 ● ライブラリ／サンプルコード（C言語・Go言語） ● 製造・検査手順書	● ファームウェア開発 ● 通信モジュール開発 ● ハードウェア開発
製造時	● SIM	● 製造ライン開発 ● 製造

※ ファームウェアのアップデート機構は、お客さまご自身で設計いただく必要があります。

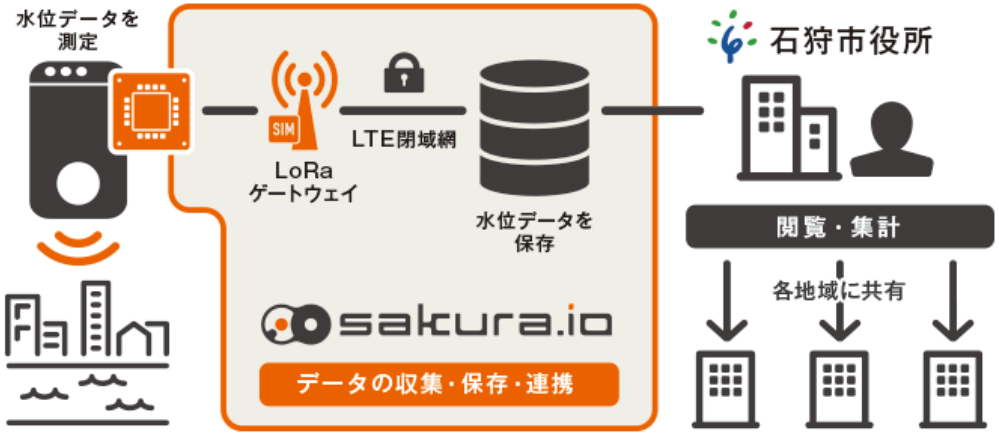
利用事例

SAKURA internet

公共
教育
農業
健康
新分野
製造



河川の水位計測

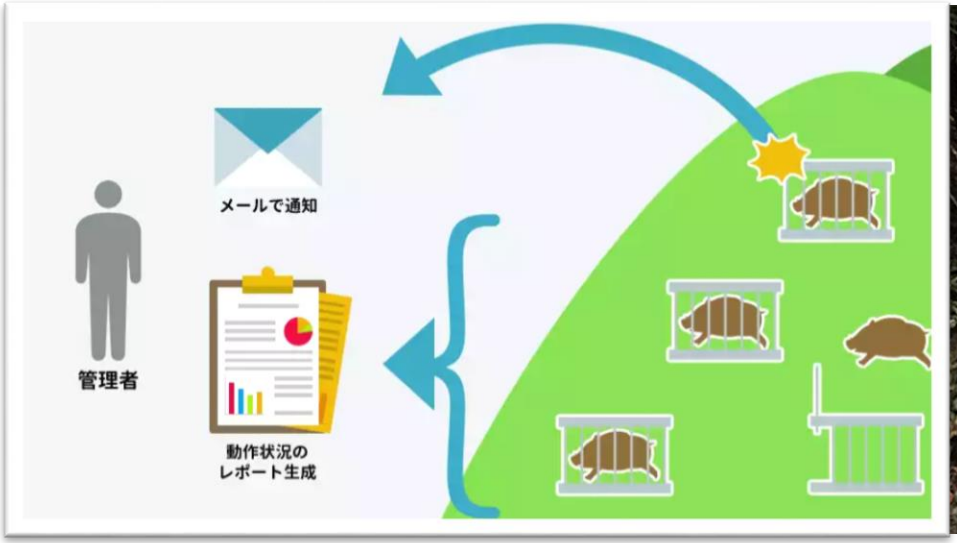


https://www.sakura.ad.jp/information/pressreleases/2018/03/30/90212/

https://sakura.io/blog/2018/10/04/eq-lpwa/

- 石狩市北部の河川(6か所)の水位計測
- 超音波センサーで測定したデータをsakura.ioに送り可視化
- LPWA採用により低消費電力化(電池で1年間連続稼働予定)

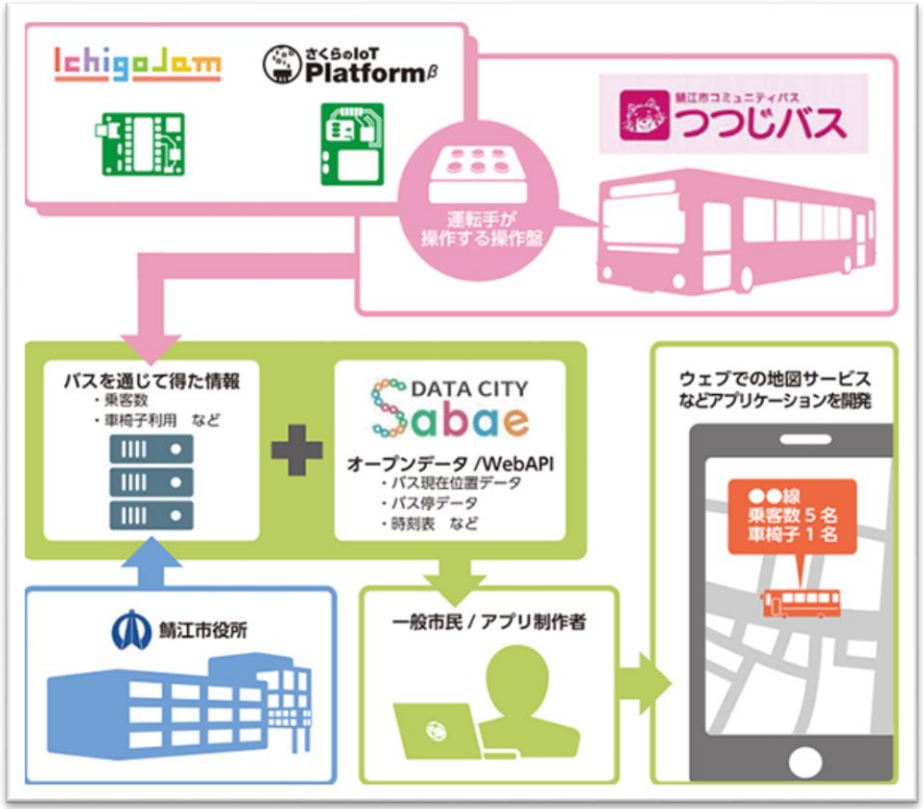
- イノシシやマングースの駆除に利用
- 罠に通信モジュールを装着し動作状況を取得
- LTEが不安定な場所ではゲートウェイを利用



http://pcn.club/katsuyama/azure201703/

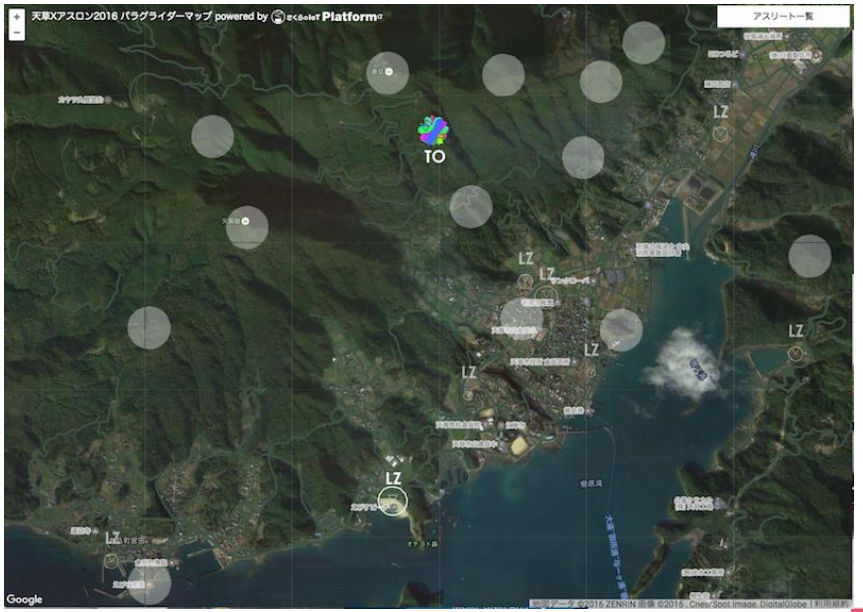
http://knowledge.sakura.ad.jp/other/7902/

- 鯖江市のコミュニティバス「つつじバス」
- 運転手の操作盤に通信モジュールを装着し乗客数などを取得
- 集計結果はオープンデータとして活用



https://www.sakura.ad.jp/press/2017/0330_sabae-tsutsujibus/

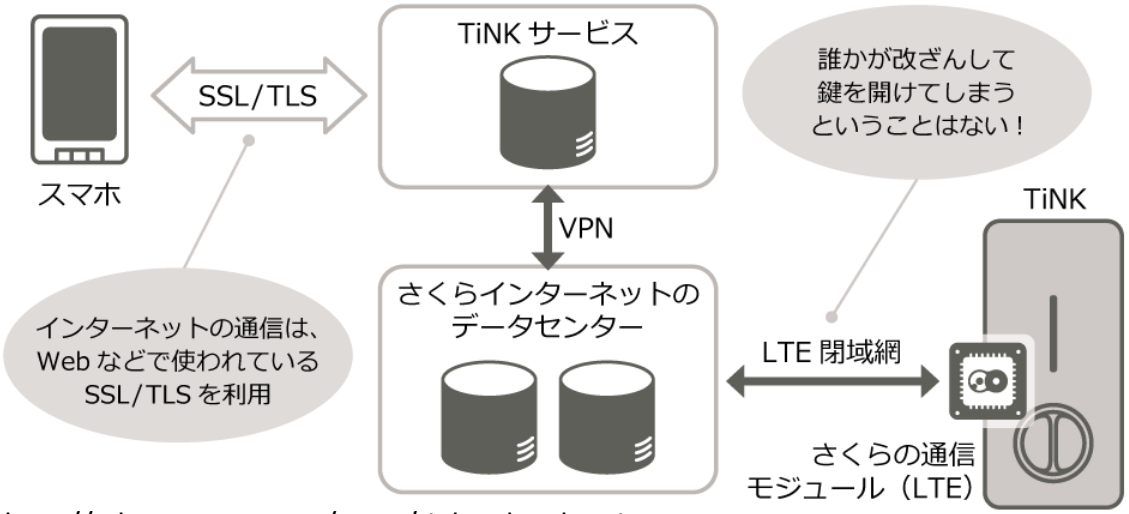
- アウトドアスポーツの競技会
- 2016年の大会にて実証実験
- パラグライダーの飛行状況をリアルタイムで表示
- 空中ではLTEが使えないのでLoRaを使用
- https://thinkit.co.jp/article/10084



スマートロック

- 3G/LTE搭載鍵デバイス「TiNK」
- 制作：株式会社 tsumug　https://www.tsumug.com/
- テンキーで暗証番号を入力しsakura.io経由で認証
- 全国の賃貸住宅への導入を目指す


merchari
How to Merchari
Debut!
Port Owners
Recr
7:00
sakura.io
products
おまたせしました、メルチャリです。
いよいよあなたの足元に。
App Store
からダウンロード



http://edge.tsumug.com/entry/tinktech-sakuraio

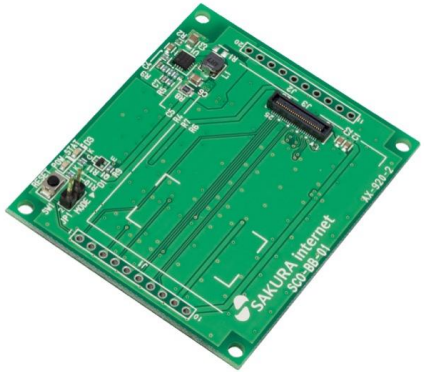
- メルカリが運営するシェアサイクルサービス
- アプリで鍵のQRコードを読み取るとsakura.io経由で解錠
- 福岡市内でサービス中


sakura.ioと
OSSの組み合わせ

電気信号とJSONを相互変換するプラットフォーム

sakura.ioの両端はオープンな通信方式とデータ形式を採用

- I2C/SPIで通信できればよい
- マイコンとはボードで接続
  - Arduino / Raspberry Pi / IchigoJam
- ライブラリやサンプルプログラムも公開
  - http://www.arduinolibraries.info/libraries/sakura-io
  - https://github.com/sakuraio
- ブレイクアウトボード(検証ボード)もあり



- sakura.ioで用意したAPIによりリクエストを送出
- 対応プロトコル/サービス
  - Websocket
  - Webhook (Incoming / Outgoing)
  - MQTT Client
  - DataStore API
  - AWS IoT
  - Azure IoT Hub / Azure Event Hubs
  - Google Cloud Pub/Sub Publisher
- URLの例
  - wss://api.sakura.io/ws/v1/(ID)
  - https://api.sakura.io/incoming/v1/(ID)
- JSON形式のデータが返ってくる
- お好みのプログラミング言語でJSONデータを処理
  - 主要な言語はJSONを扱うライブラリあり


```
{
  "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,
          ...
      }]
  }
}
```

利用例その1
温度・湿度の測定
（Node-RED編）

出典：sakura.io体験ハンズオン

http://bit.ly/sakuraio-handson-part2-01

SAKURA internet

温湿度データを取得し
sakura.ioへ送出

Websocketでデータを入手し
Node-REDで処理

マイコン
（Arduino Uno）

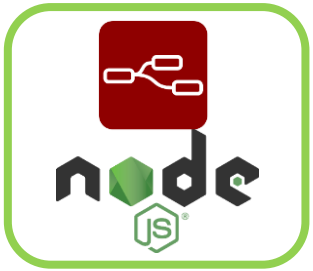
sakura.io
モジュール

温湿度センサ

LTE
閉域網

sakura.io

WebSocket



仮想サーバ
Linux（CentOS）

さくらのクラウド
sakura cloud

1. sakura.io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(Websocket)の設定
2. 機器の配線とマイコンのプログラム開発
   - 通信モジュールとArduinoシールドを接続
   - 温湿度センサーからの出力をArduinoに取り込めるよう配線
   - 温湿度情報をsakura.ioに出力するArduinoのプログラムを作成
3. サーバの作成とデータ加工処理の開発
   - Node-REDサーバを作成
   - 温湿度データを加工するフローを作成
   - WebSocketでデータを入手し、Node-REDで加工してグラフ表示＆Twitterに投稿

②
マイコンおよび
プログラムの構築

①
sakura.ioの設定

③
Webサービス連携
（さくらのクラウド）

温湿度センサ

sakura.io
モジュール

マイコン（Arduino Uno）

仮想サーバ

プロジェクトの登録

通信モジュールの登録

連携サービスの登録

連携サービスの設定

SAKURA internet

連携サービスとして、WebSocketなどのオープンな通信プロトコルや、
AWS IoTなどのようなクラウド事業者のサービスを選択することができます。

連携サービスとしてWebSocketを選択すると、WebSocketのURLが設定されます。
このURLにアクセスすると、sakura.ioとの間でJSON形式のデータを送受信します。

SAKURA internet

②

マイコンおよびプログラムの構築

①

sakura.ioの設定

③

Webサービス連携（さくらのクラウド）

温湿度センサ

sakura.io
モジュール

マイコン（Arduino Uno）

sakura.io

Node-RED

node JS

仮想サーバ

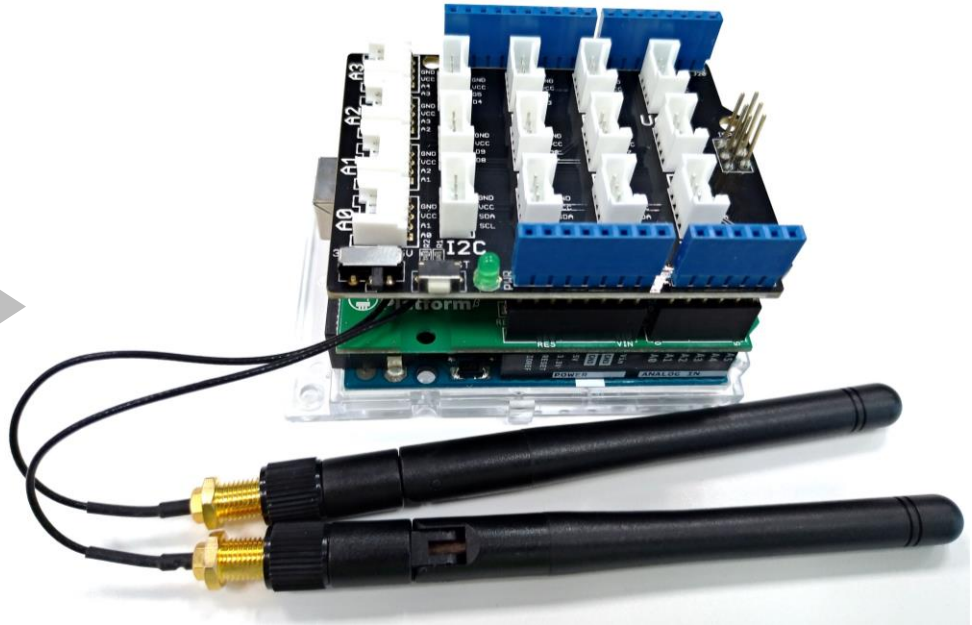
通信モジュールの下にArduinoシールドを敷き、さらにその下にArduinoを接続します。
通信モジュールにはLTEアンテナを取り付けます。

Grove ベースシールドを取り出し、ピン位置に注意しながら
組み上げ済みキットに取り付けます。

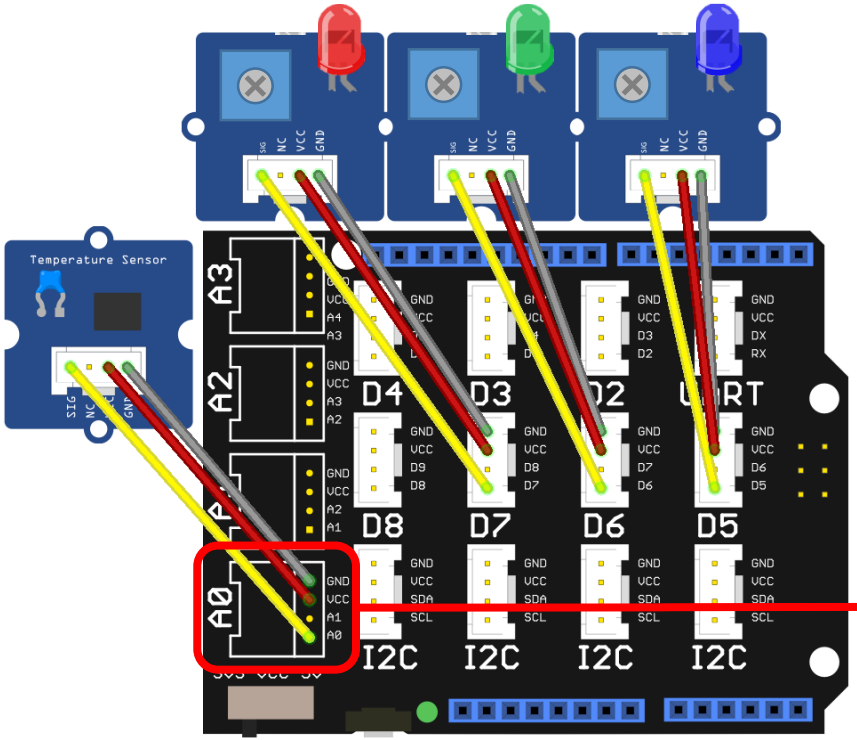
組み上げ済みキット

＋

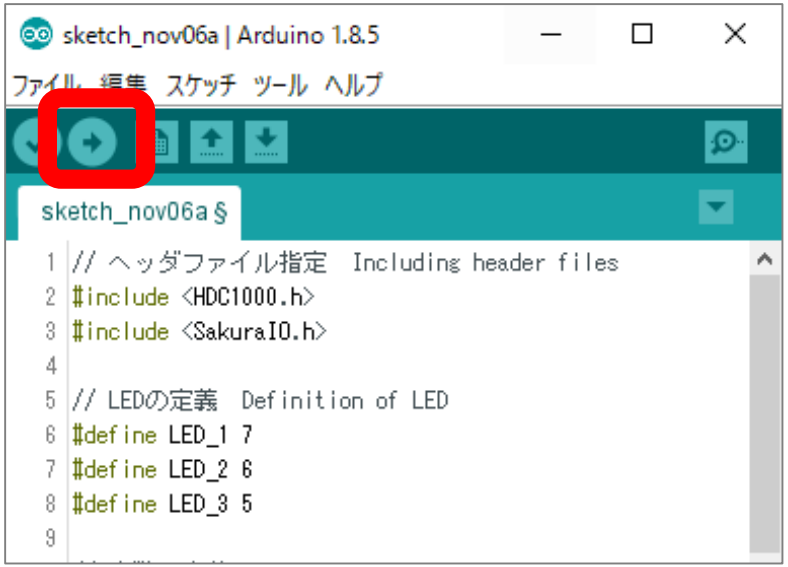
Grove ベースシールド

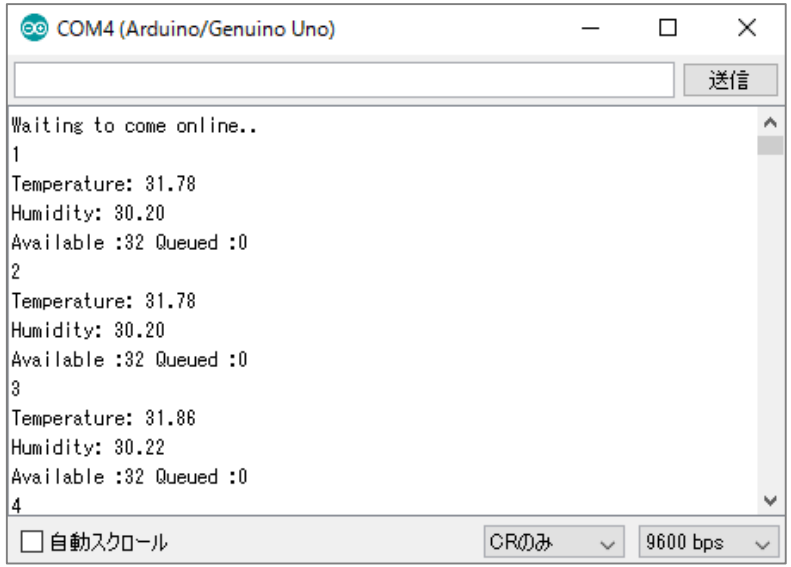


図に従い、Groveケーブルを使用して温度センサを配線します。



Arduinoにてsakura.ioを制御するためのライブラリをインストールし、
温湿度センサーの情報を送信するプログラムをArduinoに書き込みます。
プログラムが動作すると、シリアルモニタに温度・湿度・カウンタ値などが表示されます。





【参考】デバイスから送信されたデータの確認

SAKURA internet

sakura.ioの管理画面では、デバイスから送信されたデータがリアルタイムで表示されます。

サーバの作成とデータ加工処理の開発

SAKURA internet

② マイコンおよびプログラムの構築

① sakura.ioの設定

③ Webサービス連携（さくらのクラウド）

温湿度センサ

sakura.io モジュール

マイコン（Arduino Uno）

sakura.io

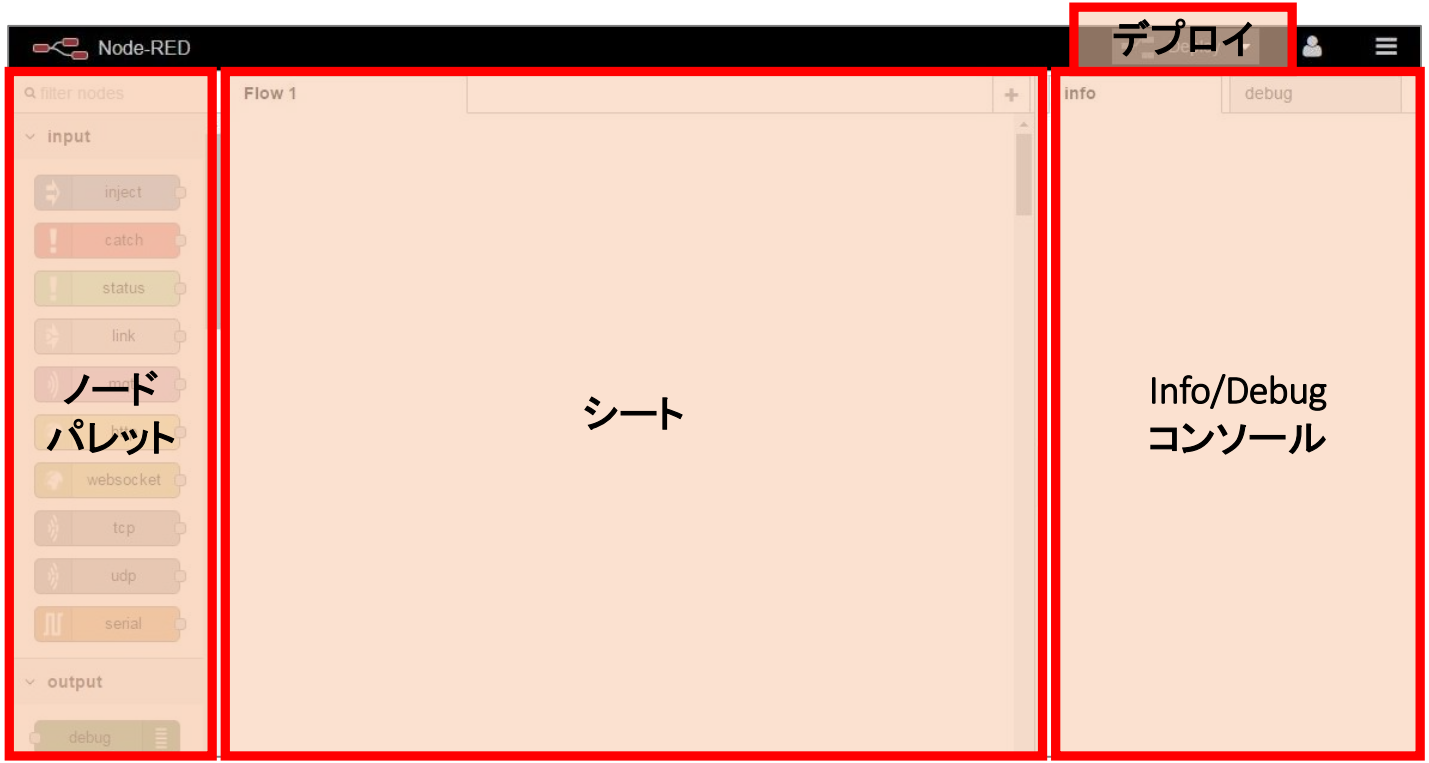
Node-RED

node.js

仮想サーバ

Node-REDとは

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です。

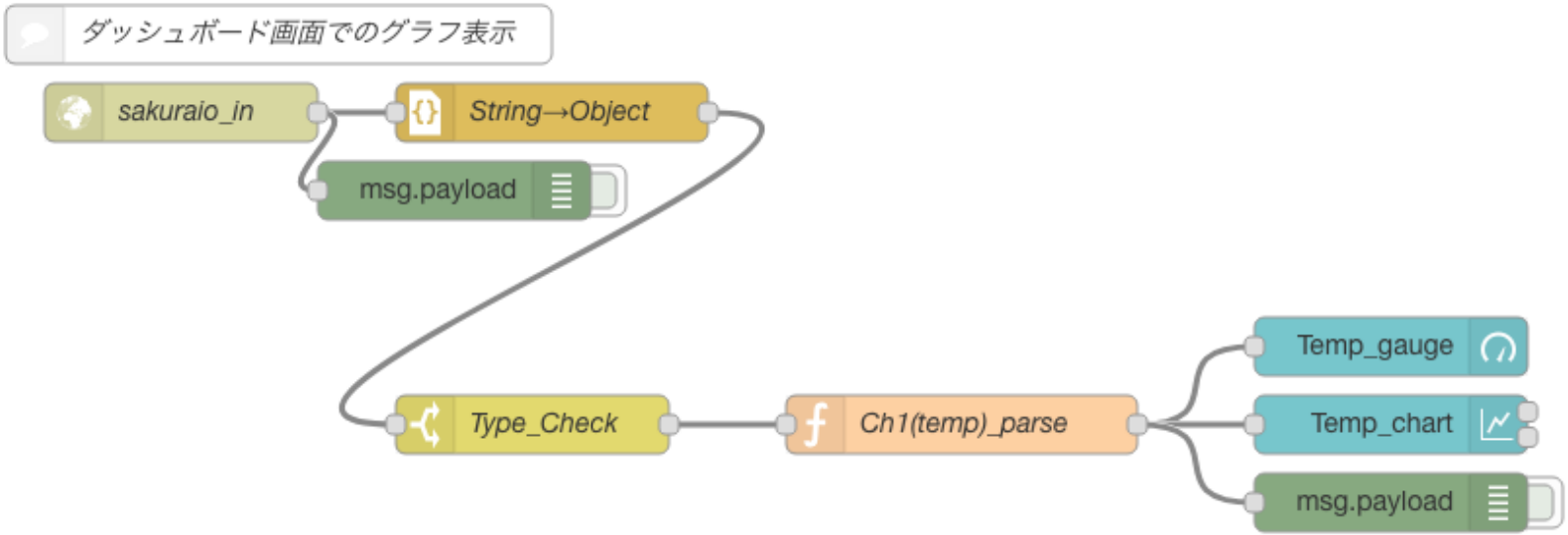


作業概要

- サーバを作成（ここではCentOS 7を使用）
- Node関連プログラムのリポジトリを登録
- Node.jsのインストール
- Node-REDのインストール
- Node-REDの自動起動設定

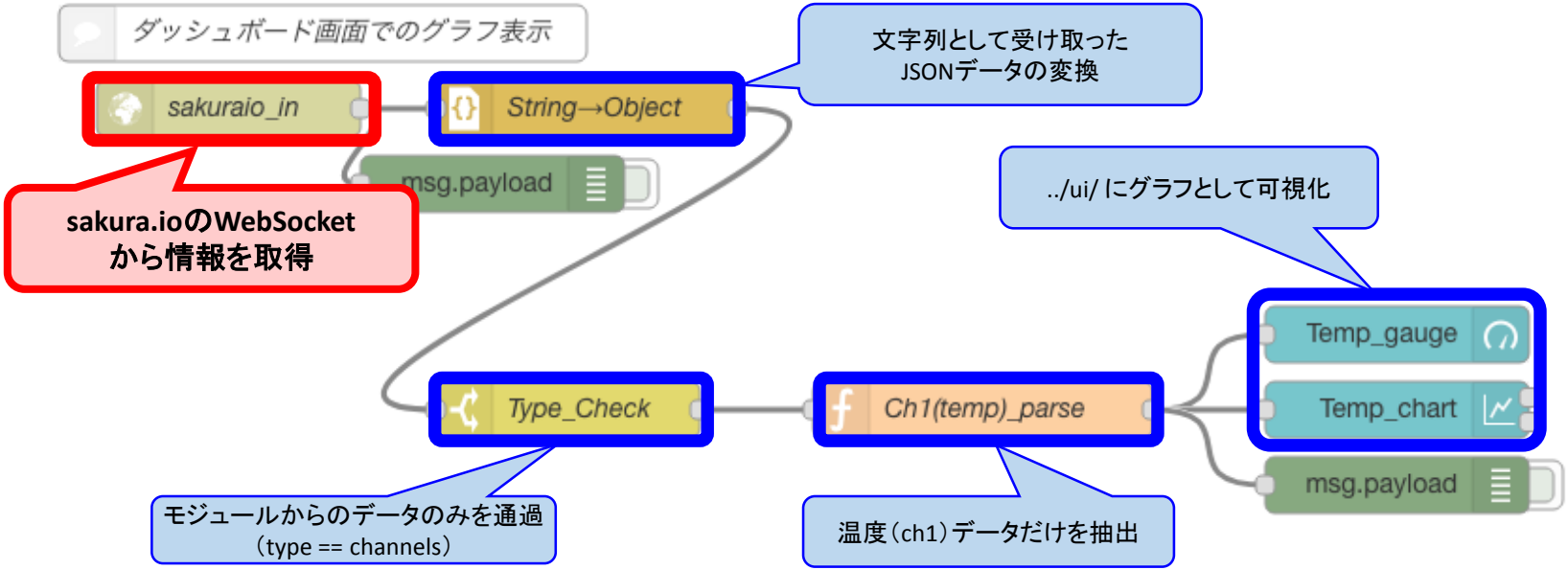
さくらのクラウドには、サーバ作成時に任意のスクリプトを自動実行する
「スタートアップスクリプト」機能があります。
スタートアップスクリプトにNode-REDを指定することにより、
前ページに掲げた作業がすべて自動的に実行され、
Node-REDサーバを簡単に作ることができます。

sakura.ioからWebSocketで温湿度データを入手し、加工して表示するフローを作成します。
WebSocketのURLを設定し、デプロイすると動作します。



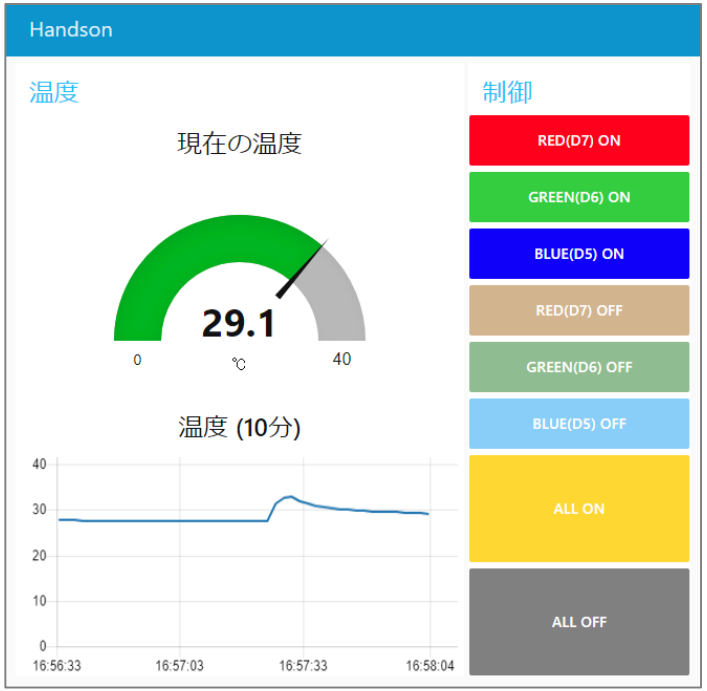
https://github.com/sakuraio/handson-sample/blob/reform/firststep/part2/nodered-firststep-part2.json

フローの内部ではこのような処理をしています。



https://github.com/sakuraio/handson-sample/blob/reform/firststep/part2/nodered-firststep-part2.json

【 http://<サーバのIPアドレス>:<指定したWeb UIポート番号>/ui/ 】にアクセスすると、
取得した情報に応じて動的にグラフが生成されることを確認できます。



利用例その2
温度・湿度の測定
（Zabbix編）

温湿度データを取得し
sakura.ioへ送出

マイコン
（Arduino Uno）

sakukra.io
モジュール

温湿度センサ

LTE
閉域網

WebSocket

Websocketでデータを入手し
Zabbixに投入

仮想サーバ
Linux（CentOS）

さくらのクラウド
sakura cloud

1. sakura.ioの設定
   - 利用例1と同じ
2. 機器の配線とマイコンのプログラム開発
   - 利用例1と同じ
3. Zabbixサーバの構築とデータ取得プログラムの開発
   - Zabbixサーバを作成
   - Zabbixにおける監視の設定
   - sakura.ioからのデータ取得プログラムを開発
   - プログラムをサーバに設置し定期的に作動するように設定

作業概要

- サーバを作成（ここではCentOS 7を使用）
- firewalldの設定
- Webサーバ(Apacheやnginxなど)の入手と設定
- DBサーバ(MySQL/MariaDB/PostgreSQLなど)のインストールとDB作成
- Zabbixのインストールと設定

Zabbixサーバの作成

さくらのクラウドには、サーバ作成時に任意のスクリプトを自動実行する
「スタートアップスクリプト」機能があります。
スタートアップスクリプトにzabbix-serverを指定することにより、
前ページに掲げた作業がすべて自動的に実行され、
Zabbixサーバを簡単に作ることができます。

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

Zabbixへのデータ投入プログラム

sakura.io からの WebSocket を受信 → 温度と湿度を zabbix_sender で送信

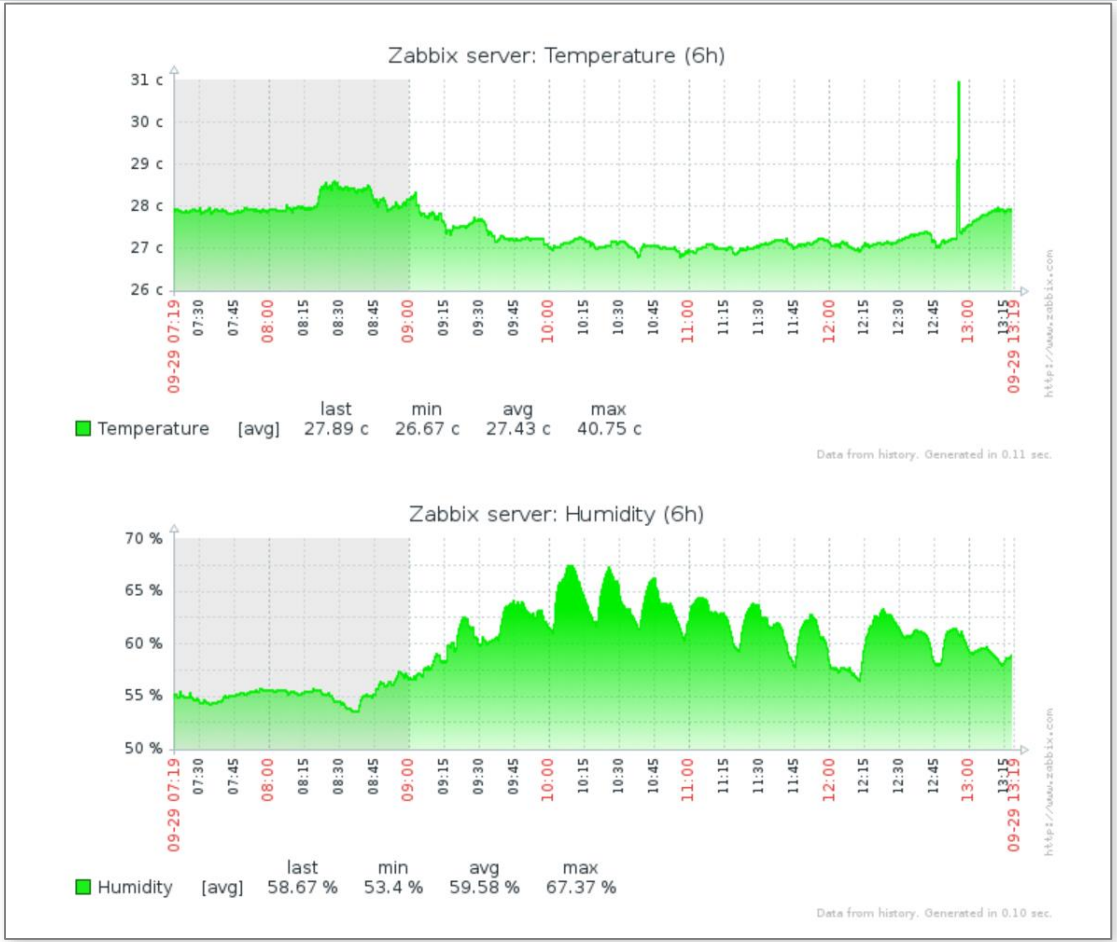
下記の例はPerlとMojoliciousを使用しているが、
WebsocketとJSONを扱えるならどのプログラム言語でも記述可能

```
#!/usr/bin/perl

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

my $ua = Mojo::UserAgent->new;
$ua->websocket('wss://api.sakura.io/ws/v1/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

SAKURA internet

温度・湿度をZabbixでグラフ化



利用例その3
LEDの操作

sakura.ioからデータを受信し
3色のLEDを点灯/消灯

Node-REDのGUIでLEDを操作し
WebSocketにてデータを送信

抵抗入りLED

sakura.io
モジュール

LTE
閉域網

sakura.io

WebSocket

Node-RED

node
JS

マイコン（Arduino Uno）

仮想サーバ

1. sakura.ioの設定
   - 利用例1,2と同じ
2. 機器の配線とマイコンのプログラム開発
   - 通信モジュールとArduinoシールドを接続
   - Arduinoからの信号がLEDに伝わるように配線
   - sakura.ioから入手したデータによりLEDを操作するArduinoのプログラムを作成
3. サーバの作成とLED操作フローの開発
   - Node-REDサーバを作成
   - LEDを操作するフローを作成しデプロイ
   - Node-REDのGUIを操作しLEDを点灯/消灯

SAKURA internet

② マイコンおよびプログラムの構築

① sakura.ioの設定

③ Webサービス連携（さくらのクラウド）

抵抗入りLED

sakura.io モジュール

マイコン（Arduino Uno）

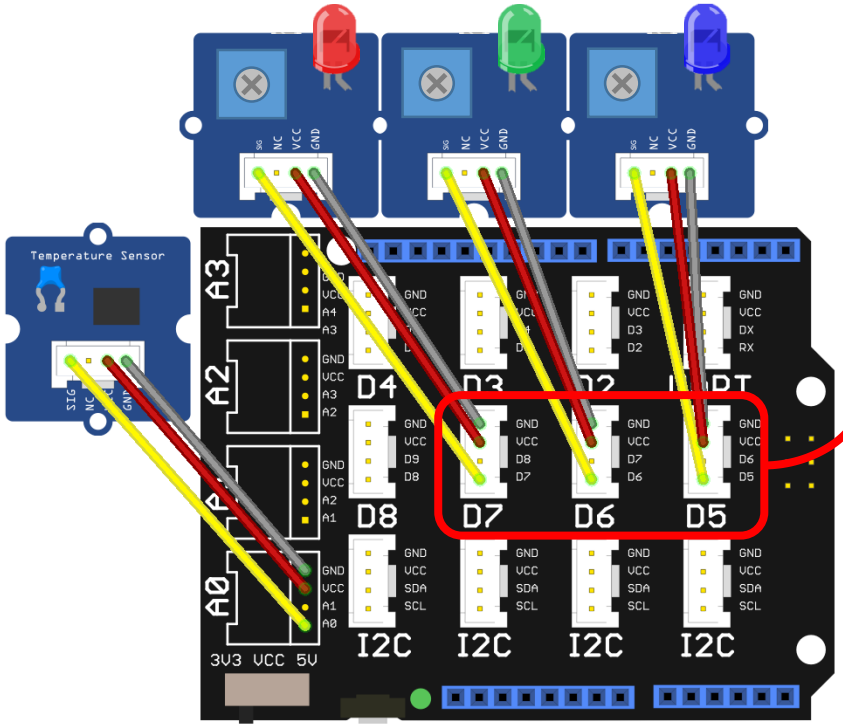
sakura.io

Node-RED

node JS

仮想サーバ

図に従い、Groveケーブルを使用してLED Socket Kitを配線します。



D7：LED（赤）
D6：LED（緑）
D5：LED（青）

sakura.ioから受信したデータに応じてLEDに信号を送るプログラムをArduinoに書き込みます。
プログラムが動作すると、シリアルモニタに結果が表示されます。
(LEDに関する情報は出ません)

SHT31 | Arduino 1.8.3

ファイル 編集 スケッチ ツール ヘルプ

SHT31

```
1  // This example requires Adafruit's SHT31 library.
2  // https://github.com/adafruit/Adafruit_SHT31
3  #include "Adafruit_SHT31.h"
4
5  #include <SakuraIO.h>
6
7  Adafruit_SHT31 sht31 = Adafruit_SHT31();
8  SakuraIO_I2C sakuraio;
```

COM4 (Arduino/Genuino Uno)

送信

```
Waiting to come online..
1
Temperature: 31.78
Humidity: 30.20
Available :32 Queued :0
2
Temperature: 31.78
Humidity: 30.20
Available :32 Queued :0
3
Temperature: 31.86
Humidity: 30.22
Available :32 Queued :0
4
```

自動スクロール CRのみ 9600 bps

サーバの作成とLED操作フローの開発

SAKURA internet

② マイコンおよびプログラムの構築

① sakura.ioの設定

③ Webサービス連携（さくらのクラウド）

抵抗入りLED

sakura.io モジュール

マイコン（Arduino Uno）

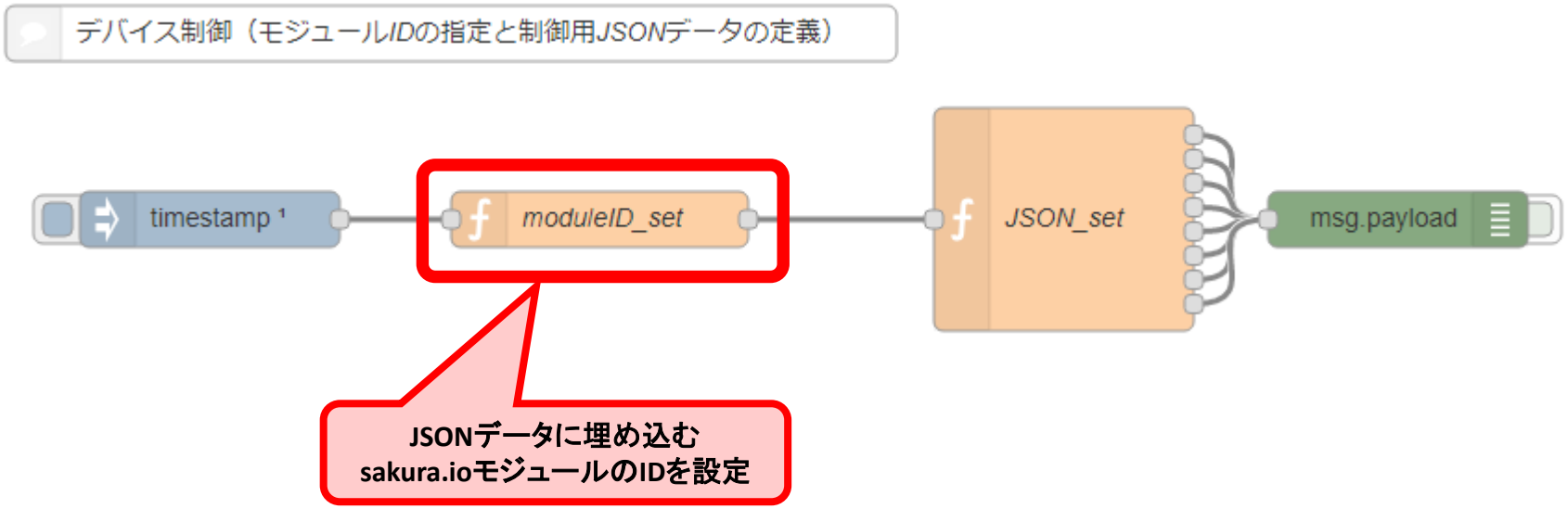
sakura.io

Node-RED

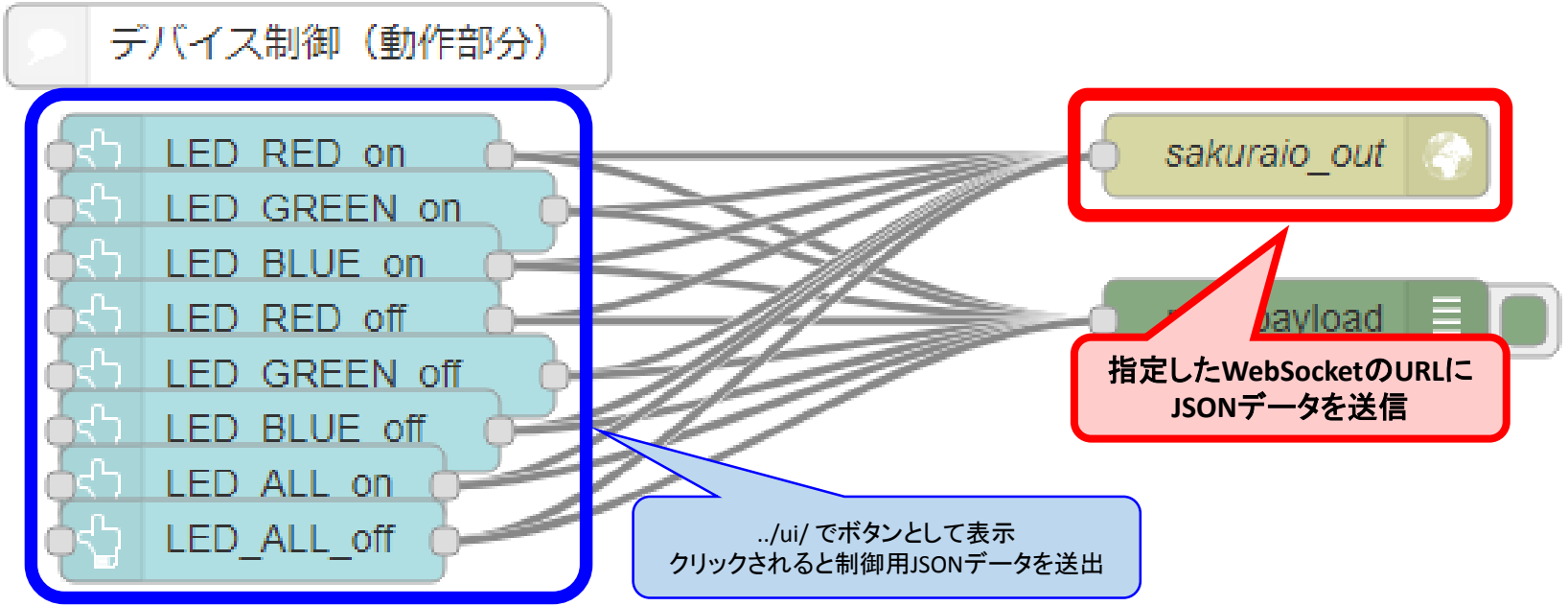
node.js

仮想サーバ

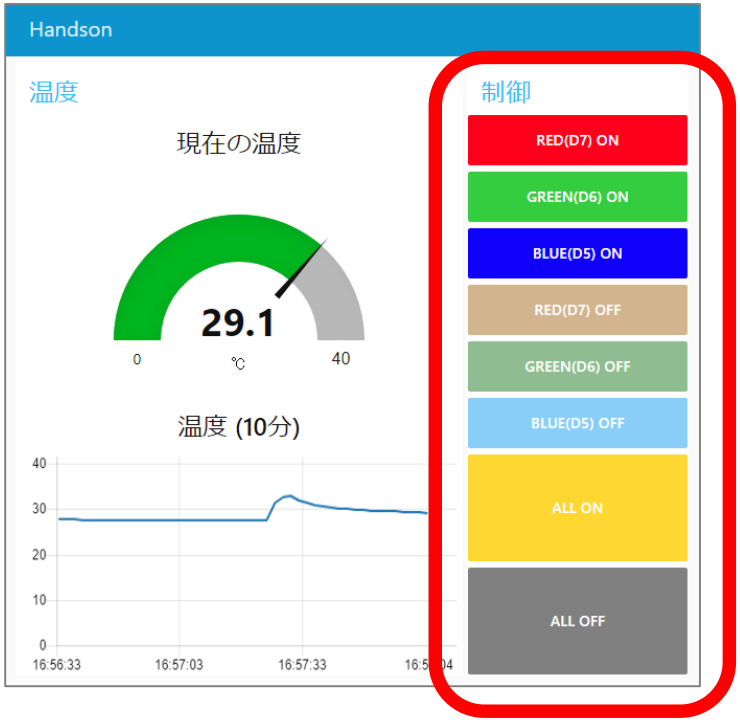
- LEDが接続されているsakura.ioモジュールのIDを指定します。
- このフローはデプロイされた時に1回だけ動作し、用途に応じた制御用JSONデータを8個作成し、それらを変数にセットします。

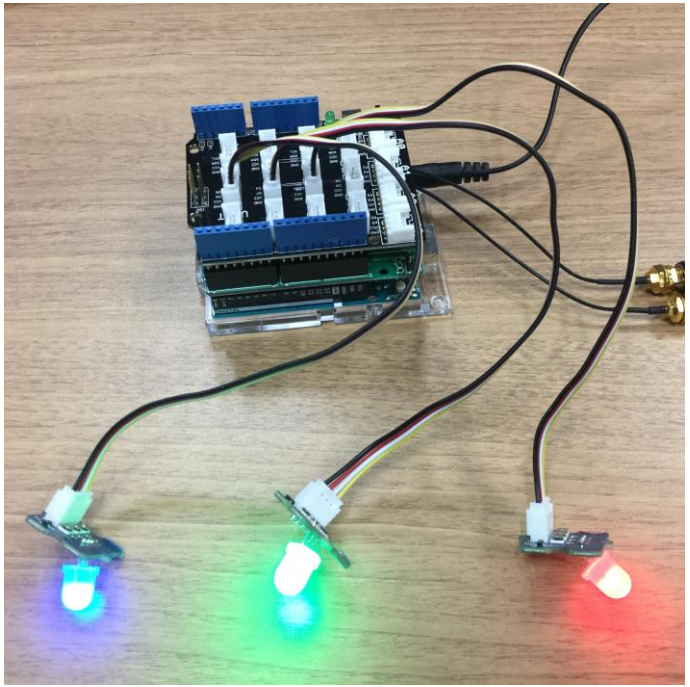


- sakura.ioからJSONデータを送信するWebSocketのURLを指定します。
- ダッシュボード画面でボタンが押下されると、制御用JSONデータを呼び出し、指定したWebSocketに送信します。



【 http://<サーバのIPアドレス>:<指定したWeb UIポート番号>/ui/ 】にアクセスし、
最右列のボタンをクリックすると、それに応じてLEDが点灯/消灯します。





まとめ

今日お話ししたこと

SAKURA internet

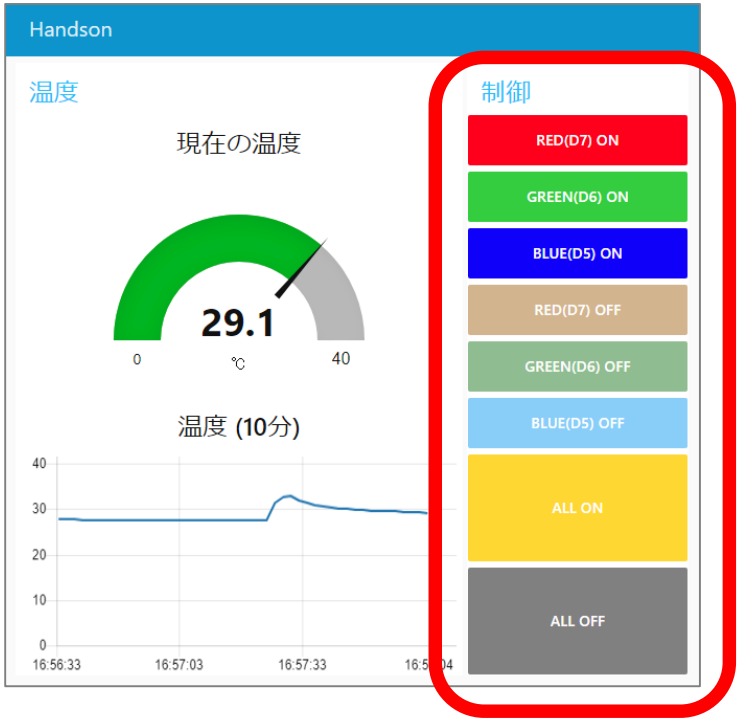
- IoTを取り巻く状況
  - データ収集方法が課題
- IoTサービスを作るときの問題点
  - 作業範囲が広範 / どの通信方式を選ぶか / セキュリティの確保
- sakura.ioについて
  - 開発経緯 / サービス概要 / 利用事例
- sakura.ioとOSSの組み合わせ
  - 温湿度センサーから取得した温度と湿度をNode-REDやZabbixで表示
  - Node-REDからデータを送信しLEDを点灯/消灯

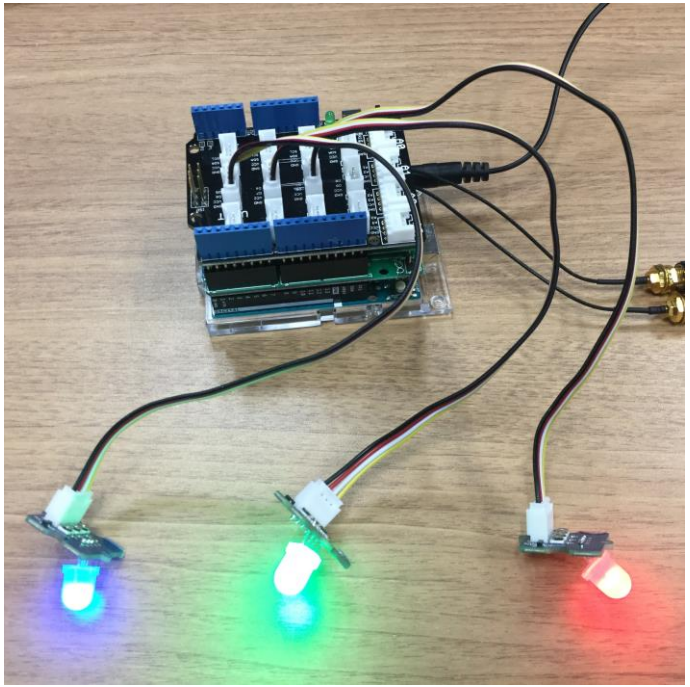
sakura.ioが目指す世界

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

参考情報

利用例(温湿度測定/LED操作)のデモと、sakura.io関連製品を展示
さくらのクラウドなどのクーポン、2019年カレンダーなども配布





- 展示ブースのデモ内容と同じことを実習
- 東京と大阪では定期的に開催中
- パートナーとの共催ハンズオンも実施中
  - Twilio / Azure / Bluemix / 駅すぱあと / AWS など
- 1月に福岡と佐賀で開催予定（1/15 福岡、1/16 佐賀）
  - 「sakura.io ハンズオン（地名）」もしくは「さくらのイベント九州版」で検索
- 他県でも開催したい！協力者求む！

- sakura.ioを使ったIoT工作の解説書
- 2018年2月発売
- 主な内容
  - sakura.ioの使い方
  - Arduinoでsakura.ioを使う
  - Raspberry Piとsakura.ioの接続
  - LinuxやPythonでセンサーの値を処理
  - sakura.ioから受信したデータの処理（JavaScript、MQTT、データストアなど）

全国でさくらのイベントを！

- さくらのイベントを全国で開催したい！
  - sakura.ioのハンズオン
  - さくらのクラウドなど各種サービスのハンズオン
  - 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
  - 連絡先：sakura-club@sakura.ad.jp

そこに、さくら