

さくらのIoTプラットフォーム 「sakura.io」を使ってみよう

<https://www.sakura.ad.jp/>

DAY

2018/2/23

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

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 #osc18tk



 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)
 - **[New!]** 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

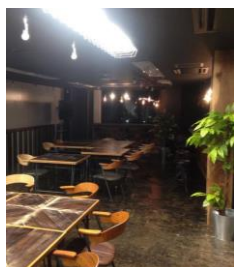
- IoTを取り巻く状況
- IoTサービスを作るときの問題点
- sakura.ioについて
 - 開発経緯
 - サービス概要
 - 利用事例
- sakura.ioとOSSの組み合わせ



大阪本社(梅田/大阪)



東京支社(西新宿)



福岡オフィス(赤坂)



東証一部上場



10821461(04)



ICMS-SP006JIS Q.27001 (ISO/IEC 27001)

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	株式会社Joe'sクラウドコンピューティング ゲヒルン株式会社 株式会社S2i アイティーエム株式会社 櫻花移動電信有限公司 ビットスター株式会社

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

1台のサーバを複数の契約者でサーバを共有または占有することができ、管理はさくらインターネットに任せて使うサービス

1台を共有

1台を占有



VPS・クラウド



さくらのVPS



さくらのクラウド
SAKURA CLOUD

仮想化技術を用い、1台の物理サーバ上に複数の仮想サーバを構築し、仮想専用サーバとして分けた領域の占有サーバ

高性能サーバと拡張性の高いネットワークを圧倒的なコストパフォーマンスで利用できるIaaS型/パブリッククラウド・サービス

専用サーバ



さくらの専用サーバ
SAKURA DEDICATED SERVER

高性能で拡張性と信頼性の高いサーバをまるごと独占して利用することができ、自由にカスタマイズして利用可能なサービス

1台～複数台



データセンター



ハウジング
リモートハウジング

データセンター内にお客様専用のハウジングスペースを確保し、ネットワーク機器やサーバなどの機材を自由に置けるサービス

新サービス

IoT



IoTデバイスからのセンサデータ蓄積や活用のため、モジュールを含めて一体型で提供するIoTプラットフォーム・サービス

さくらのセキュアモバイルコネク

クラウドにダイレクトに接続し、任意のネットワークへ接続可能なSIM。セキュリティを担保し、国内で最も安価な通信を提供

AI・人工知能



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

【サービスの主な利用用途】

ウェブサイト運営、ブログ、インターネット・メール

ネットビジネス、電子商取引、動画・音楽配信、開発環境

エンタープライズ

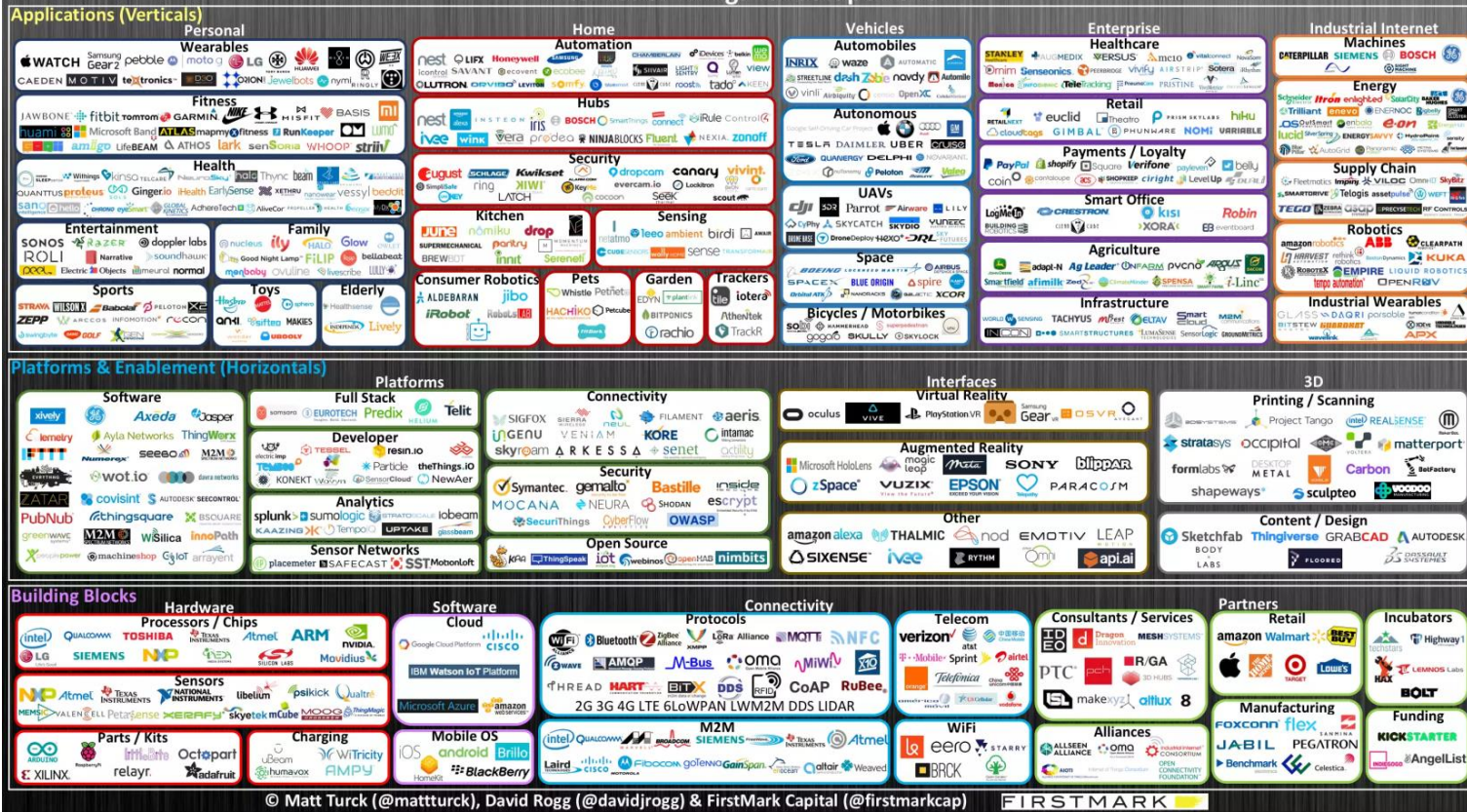
会員制サイト、キャンペーン・サイト

SNS、ウェブ・アプリケーション、SaaS、ASP

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

Internet of Things Landscape 2016



© Matt Turck (@mattturck), David Rogg (@davidjrogg) & FirstMark Capital (@firstmarkcap) FIRSTMARK

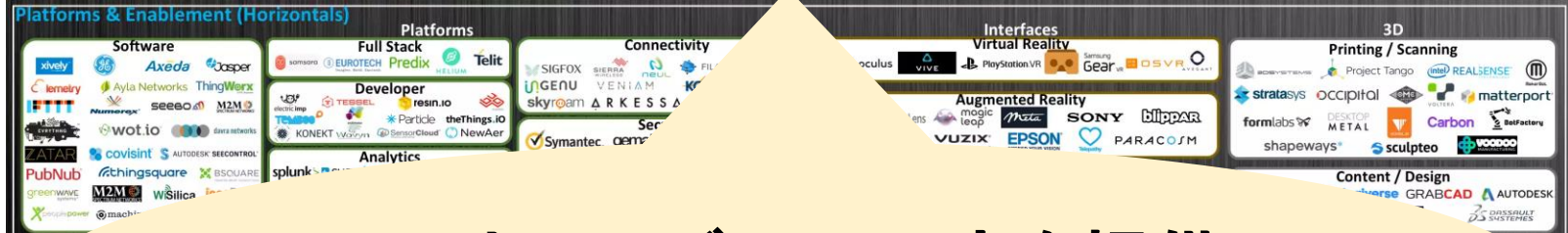
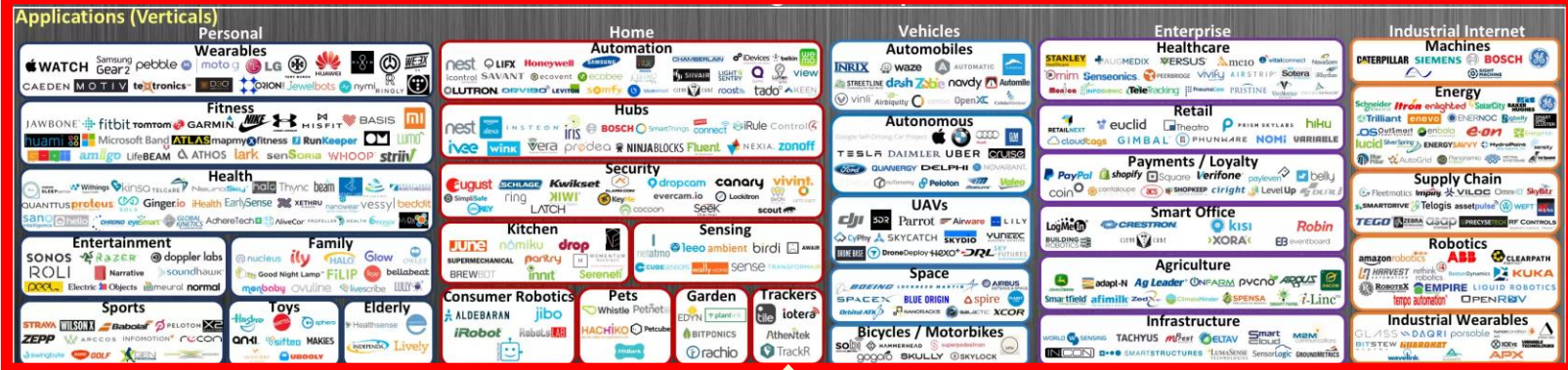
Internet of Things Landscape 2016



“インターネット”と同じぐらい広すぎる



Internet of Things Landscape 2016



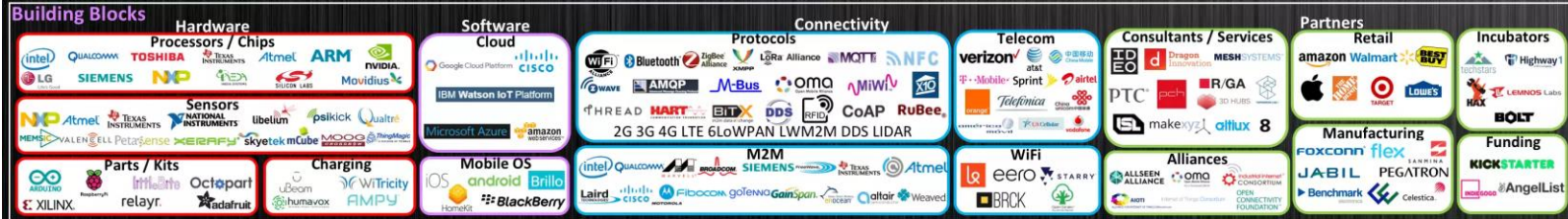
モノとサービスの両方を提供
(垂直統合型)



Internet of Things Landscape 2016



汎用的な基盤を提供
(水平統合型)



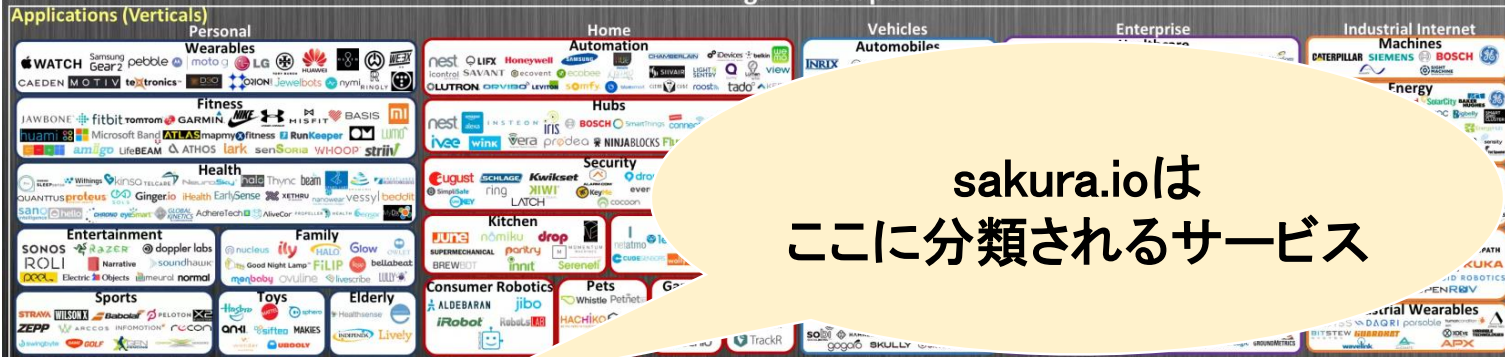
Internet of Things Landscape 2016



各種構成要素を提供
(要素技術)

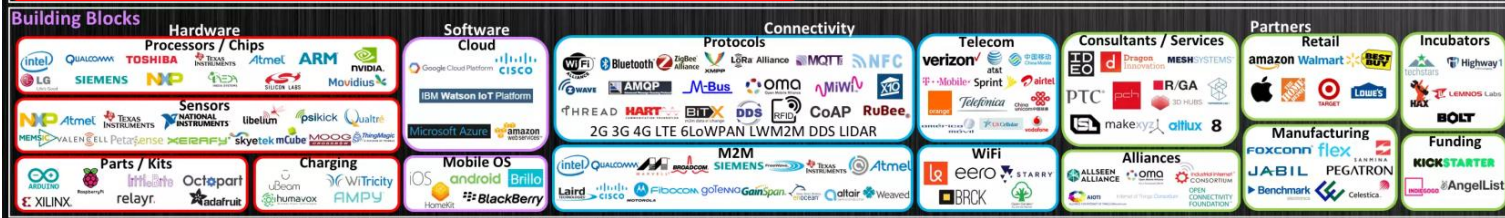


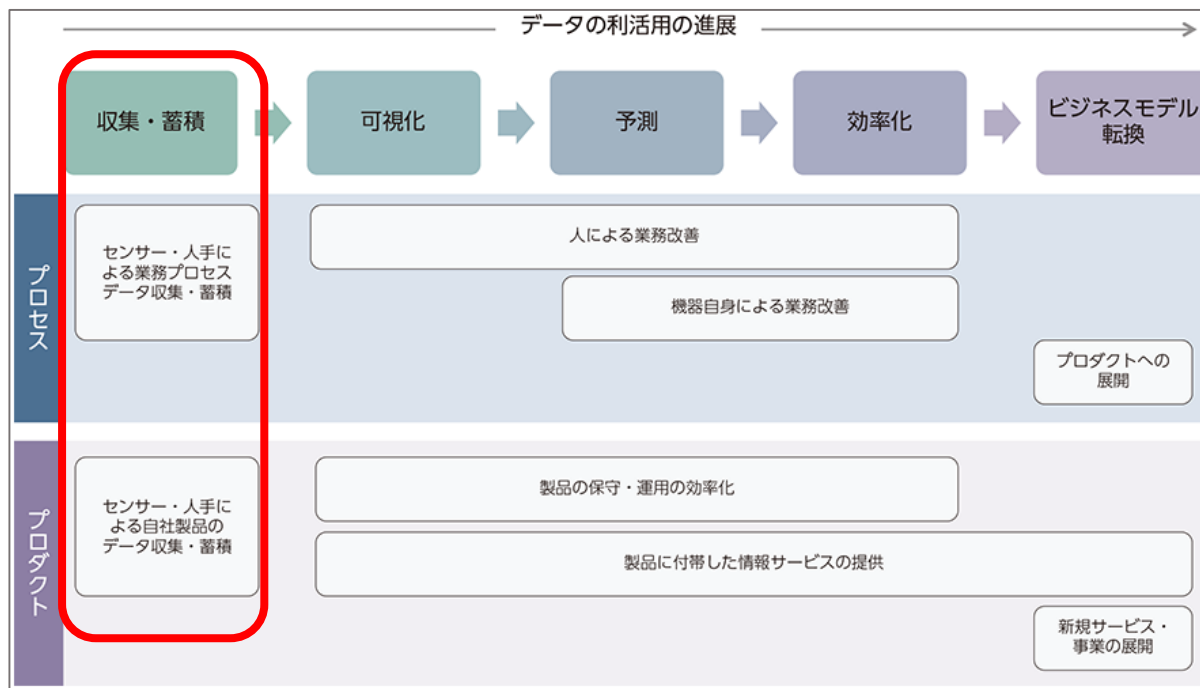
Internet of Things Landscape 2016



sakura.ioは
ここに分類されるサービス

ソフトウェア/接続性/分析/セキュリティ
などのプラットフォームサービス





出典：総務省 <http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h28/html/nc123200.html>

IoTによりデータが収集・活用されることで
ビジネスモデル転換を生み、さらにデータの活用が促進される

どうやって
データ
集めるの？

Internet of Things Landscape 2016

Applications (Verticals)

Personal Wearables Apple WATCH, Samsung Gear, Pebble, LG, Huawei, Fitbit, Jawbone, etc.	Home Automation Nest, LIFX, Honeywell, Philips Hue, etc.	Vehicles Automobiles NIRX, etc.	Enterprise IBM, etc.	Industrial Internet Machines Caterpillar, Siemens, Bosch, etc.
Fitness Fitbit, Garmin, etc.	Hubs Nest, Philips Hue, etc.			Energy etc.
Health Fitbit, etc.	Security Kwikset, etc.			
Entertainment SONOS, etc.	Kitchen Philips Hue, etc.			
Sports Fitbit, etc.	Pets Whistle, etc.			

sakura.ioは
「どうやってデータ集めるの？」
に挑戦するサービス

Platforms & Enablement (Horizontals)

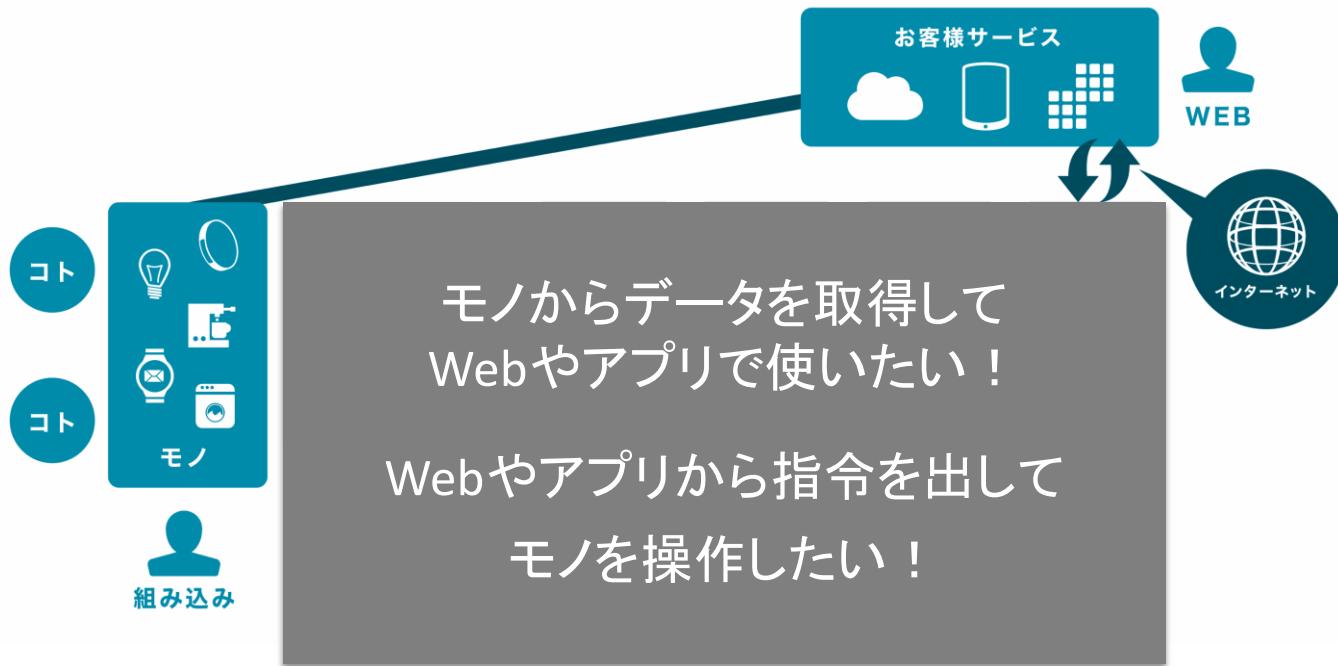
Virtual Reality Oculus, Vive, PlayStation VR, etc.	3D Printing / Scanning Formlabs, etc.
Augmented Reality Microsoft HoloLens, etc.	Content / Design Sketchfab, etc.
Other Amazon Alexa, etc.	

ソフトウェア/接続性/分析/セキュリティ
などのプラットフォームサービス

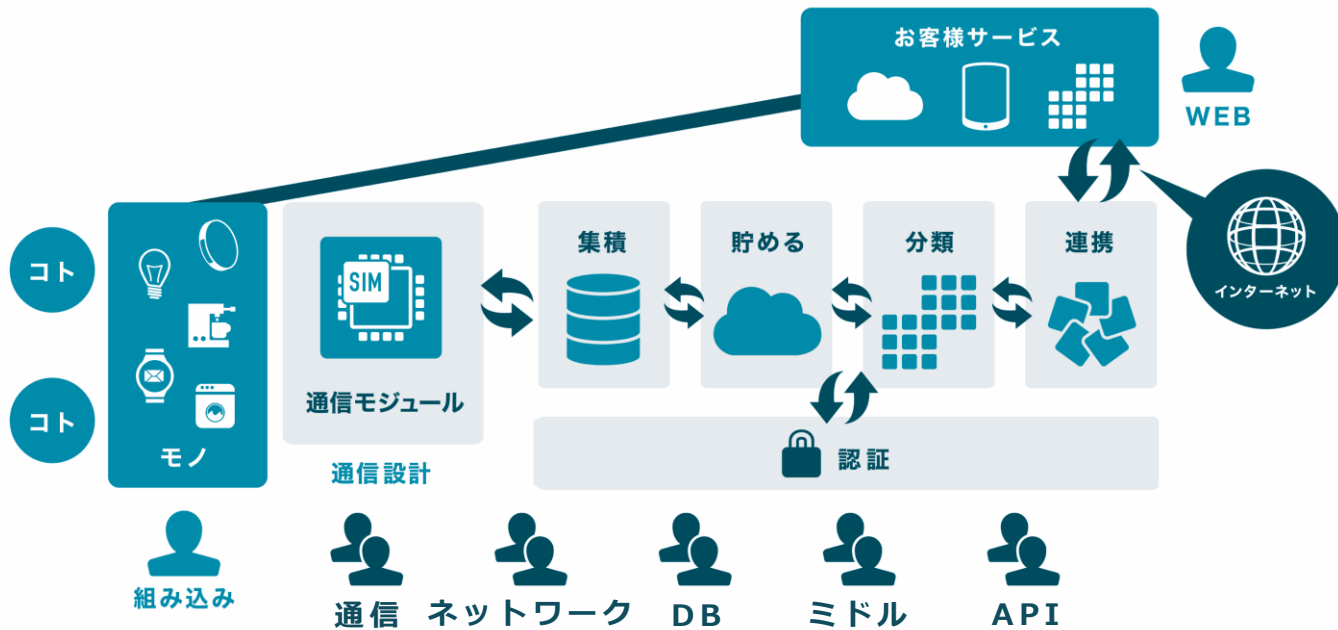
Building Blocks

Hardware Processors / Chips Intel, Qualcomm, Toshiba, ARM, etc.	Software Cloud Google Cloud Platform, IBM Watson IoT Platform, etc.	Connectivity Protocols WiFi, Bluetooth, ZigBee, etc.	Telecom Verizon, AT&T, etc.	Consultants / Services PTC, etc.	Partners Retail Amazon, Walmart, etc.	Incubators Highway1, etc.
Sensors Texas Instruments, etc.	Mobile OS iOS, Android, etc.	M2M Siemens, etc.	WiFi Intel, etc.	Alliances AllSeen Alliance, etc.	Manufacturing Foxconn, etc.	Funding Kickstarter, etc.

IoTサービスを 作る時の問題点



モノをインターネットにつないで何かしたい



ハードウェア/ソフトウェア両方の知識が必要

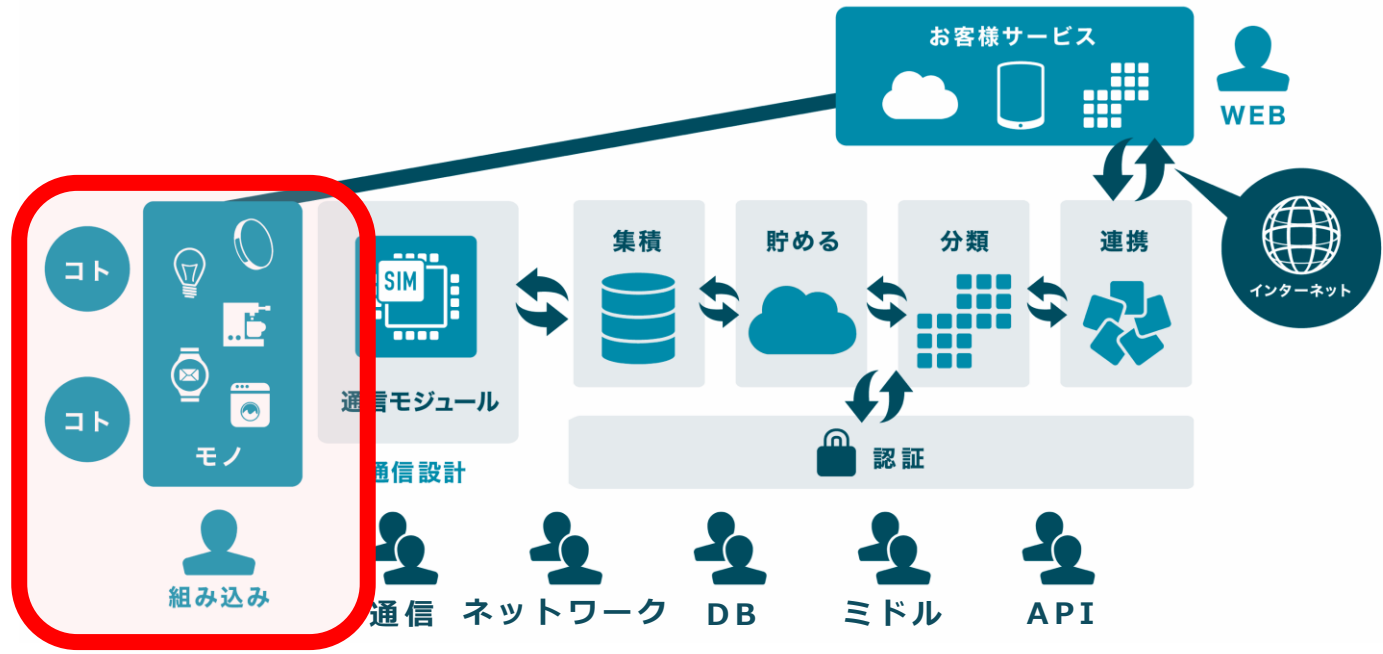
IoTサービスを作るときに必要な作業

- 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

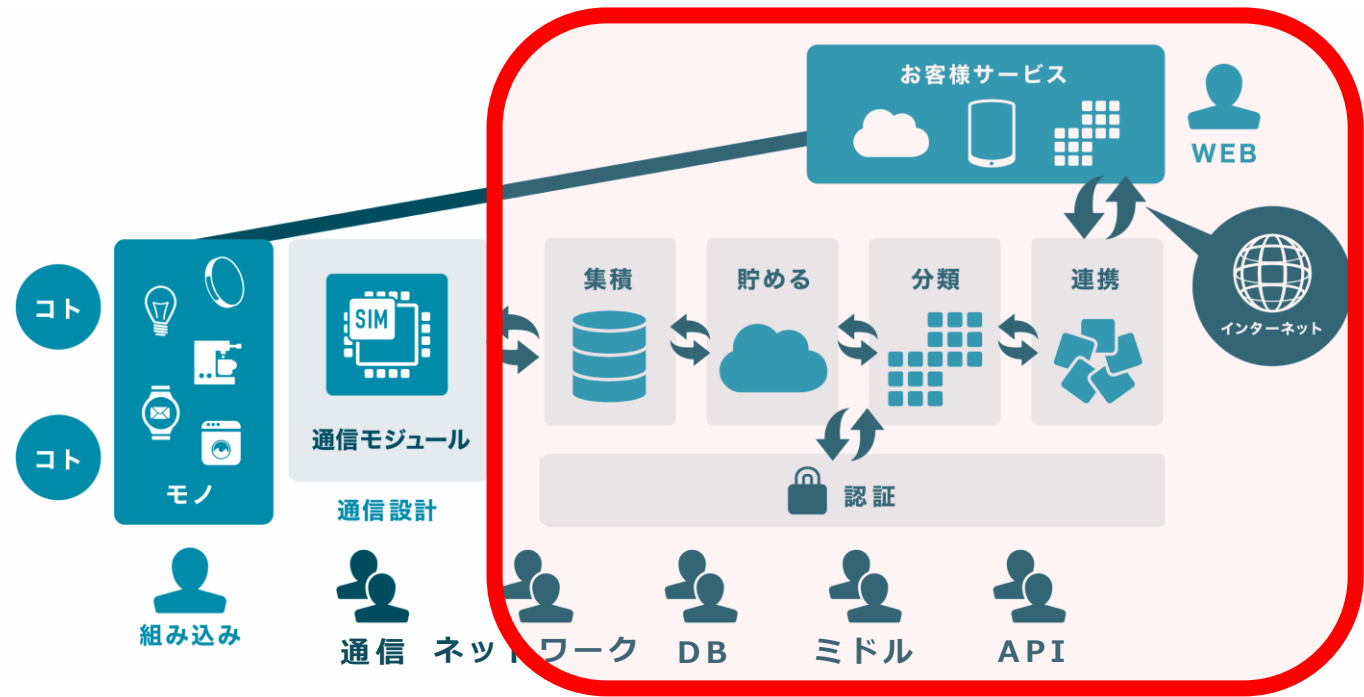
IoTサービスを作るときに必要な作業

これらを全部できる人は
ほとんどいない

- セキュリティ
 - ユーザ認証 / 機器認証 / 暗号化 / バックアップ











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

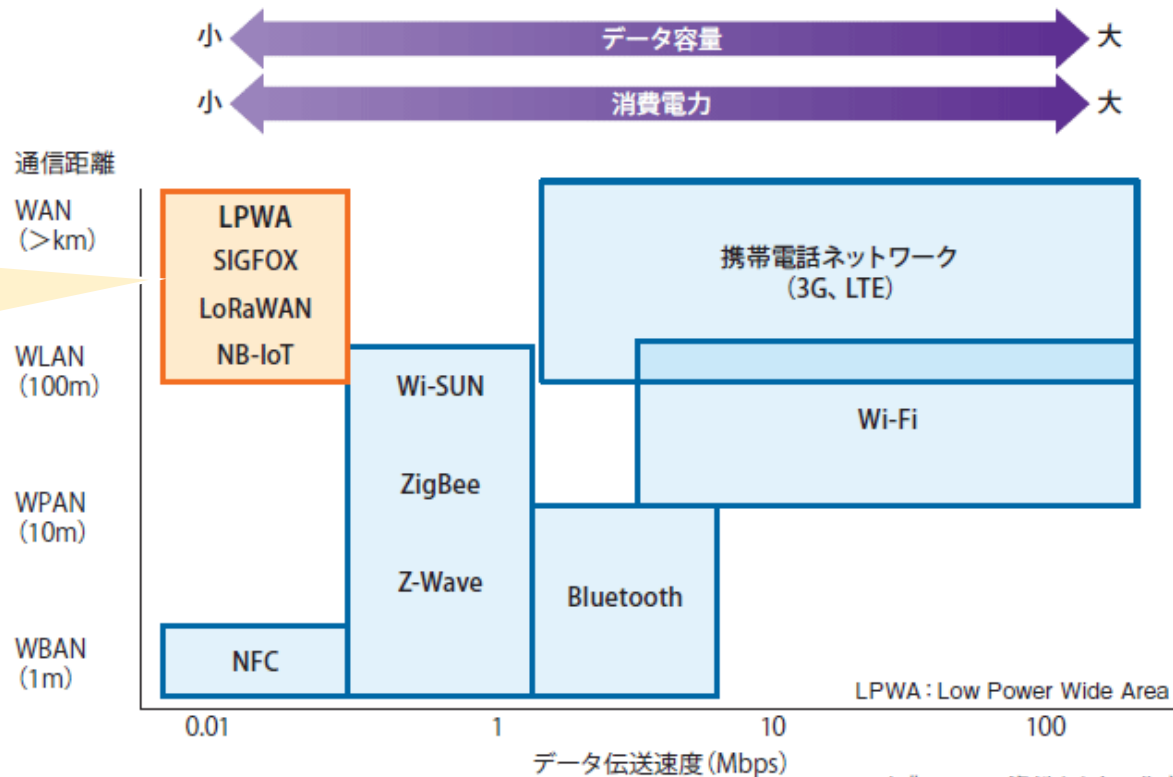


TCP/IPやHTTPのことはわかるが
物理的なことや電気的なことはわからない

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	 	  
どこでもつながる		 

IoT向けの
通信方式
として注目



出典 : KCCS資料をもとに作成

どの方式にも弱点がある

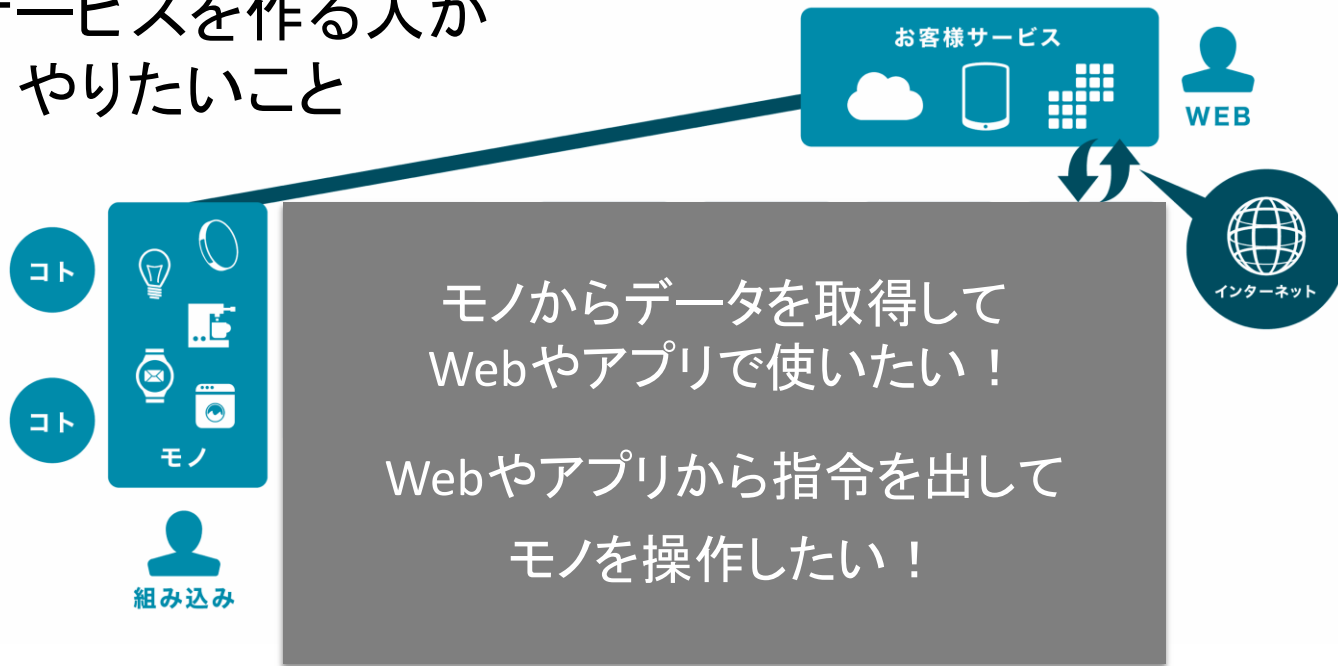
- LAN / WiFi: 回線やアクセスポイントが必要
 - モノをつなぐためだけにBフレッツを敷設？
- LTE
 - 山奥や地中深くでは電波が入らない
 - 空中では使用禁止(電波法)
- Bluetooth: 近距離でないとならぬ
- その他: 普及していない

- 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がると攻撃対象になる
 - 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

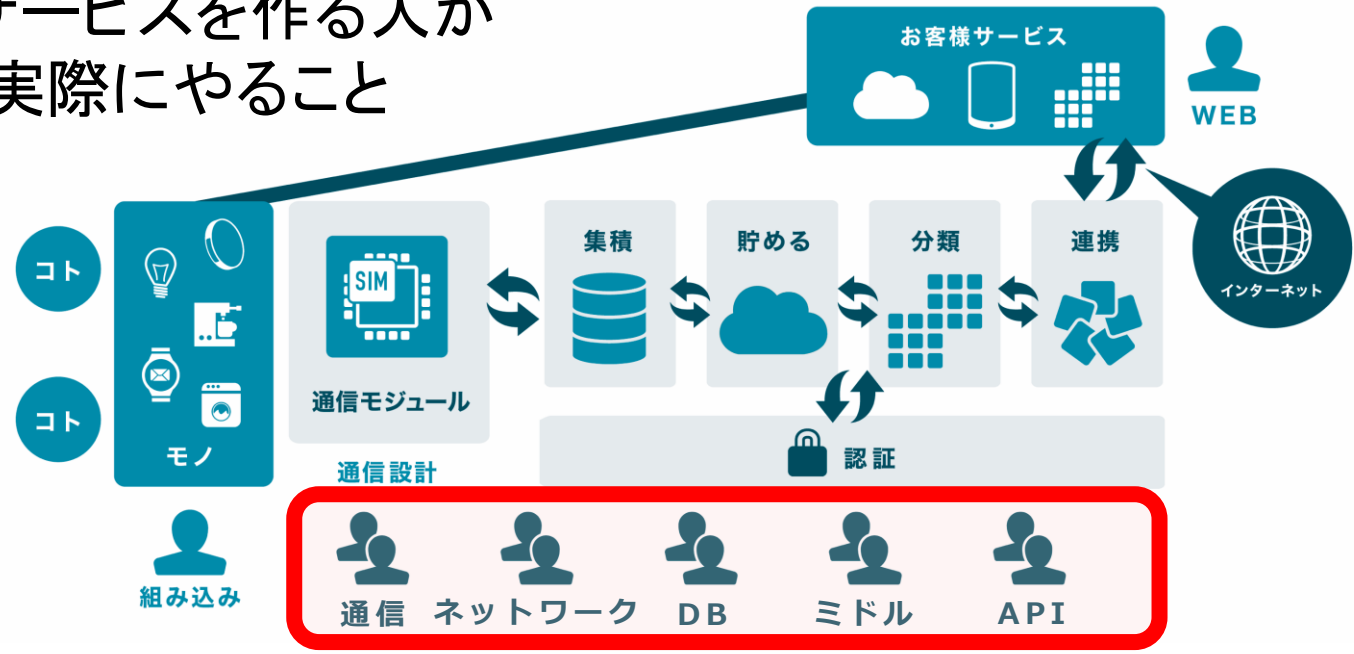
さくらインターネットの IoTへの取り組み

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

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い



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概要

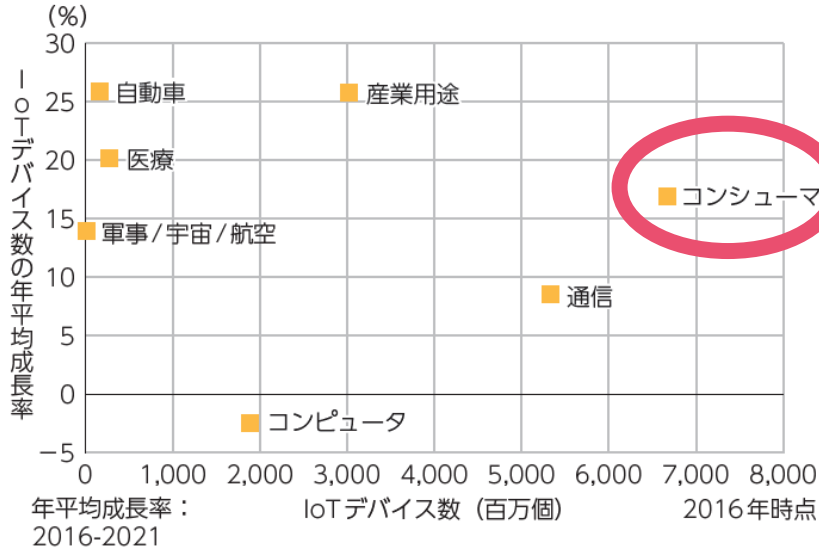


既存の事業領域 や エンジニアのスキルセット を
変更することなく、モノやサービス作りに注力することができる
かつ、収集されたデータの流通 も見据えた基盤

これまで気付けなかった「モノ・コト」の相関性や関係性を見出し、
それを世界でシェアできるプラットフォーム がコンセプト

図表3-3-1-2

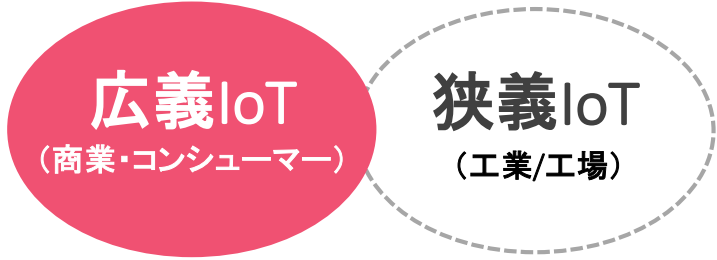
分野・産業別のIoTデバイス数及び成長率



(出典) IHS Technology

出典:総務省 <http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h29/pdf/29honpen.pdf>

「sakura.io」のターゲット分野



低単価だが母数が多い【**広義のIoT**】領域に注力

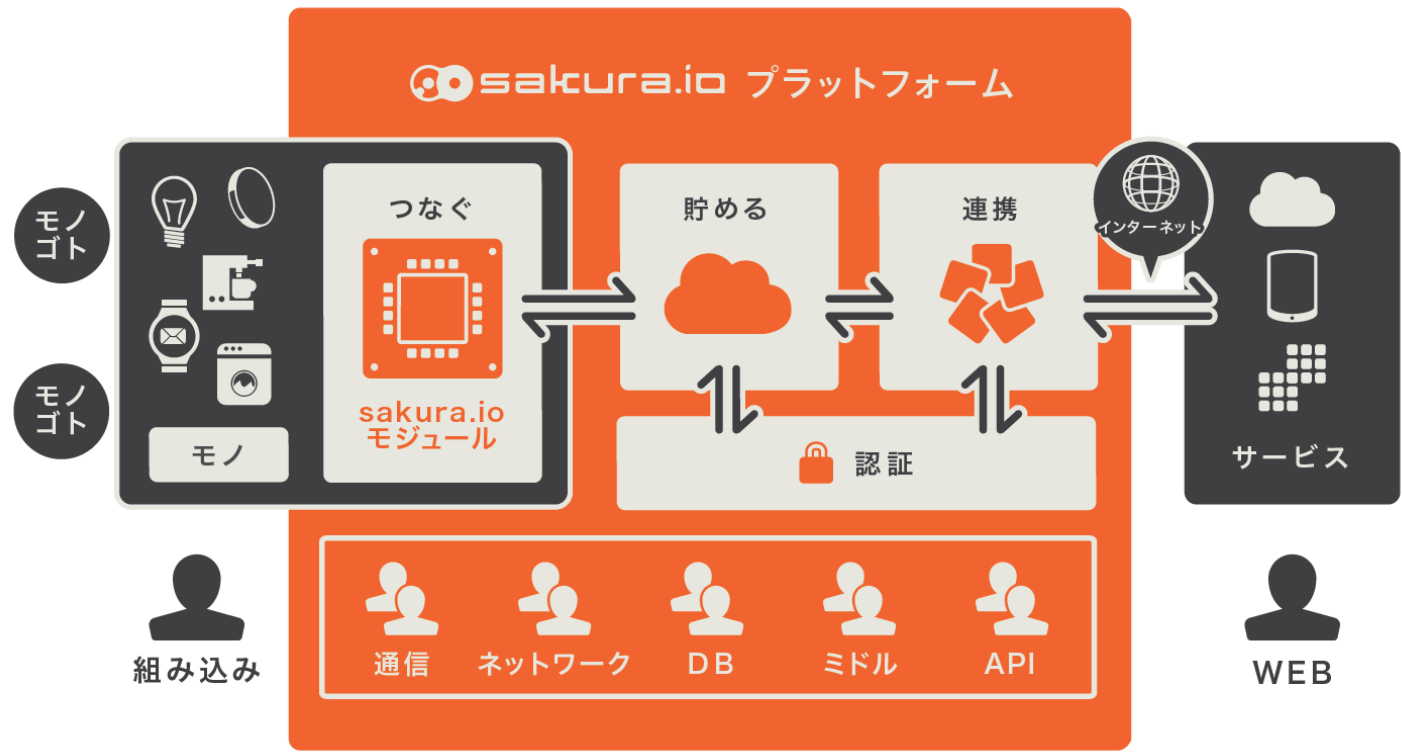
✓ デバイス数**60億個**以上

(2016年時点)

✓ 成長率**15%**以上

(2016-2021年予想)

サービスとしては母数が多く、成長が期待されている
「コンシューマ」分野に携わる「法人」にフォーカス(所謂B2B2Cモデル)

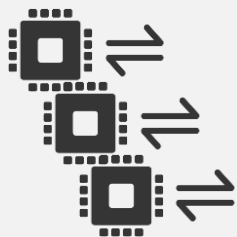


【モノ】と【データセンター】を直結することでデータ活用を促進
どちらの業種にも新技術習得や事業領域変更は不要

		sakura.io	他社IoTプラットフォーム	IoT用MVNO事業者
やりたい	企画・アイデア	<div style="border: 2px solid red; padding: 10px;"> <p>「データを迎えに行く」という発想</p> <ul style="list-style-type: none"> ・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 ・電源を入れるだけで利用可能 </div>		
	センサー/チップ			
	デバイス (外装、基板、ファームウェア)			
	サービス (可視化/予測/効率化)		●	
やらねばならぬ	データの送受信手段	●	○	○
	安全な通信経路	●	○	●
	デバイス認証/管理	●	●	●
	プラットフォーム機能 (収集/蓄積/連携)	●	●	○

【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 詳細



データの収集



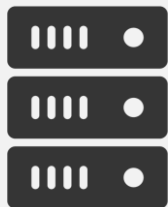
データの蓄積



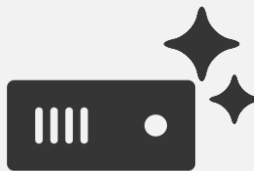
データの連携



運用機能



ラージスケール対応



アップデート

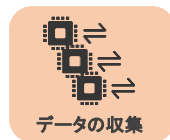


障害切り分け、復旧



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



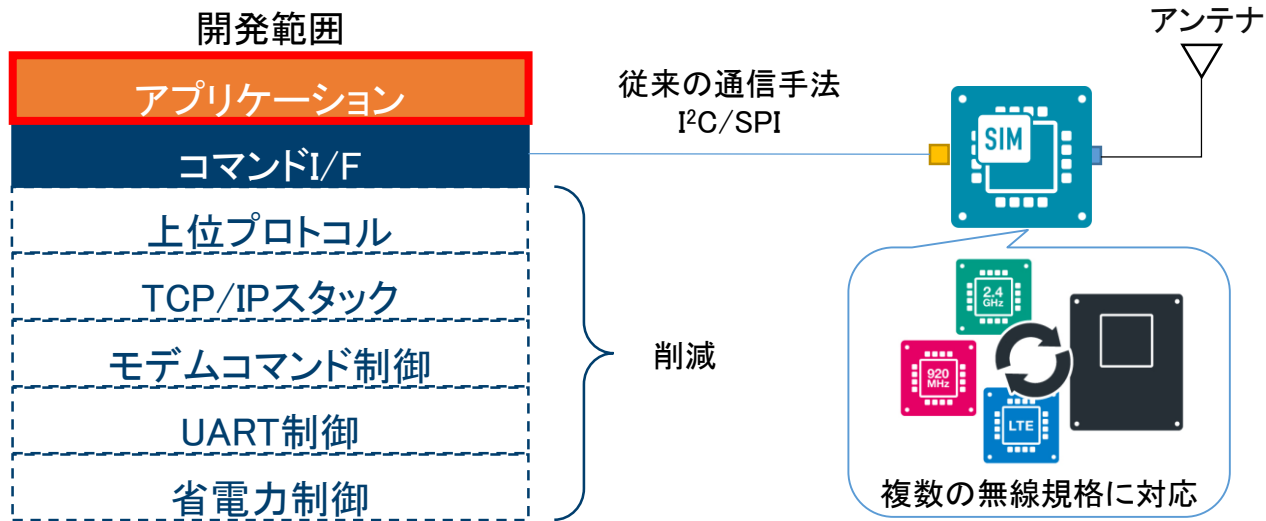
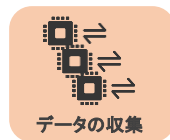
LTEモデムチップ

通信制御用MCU

SIMカードコネクタ



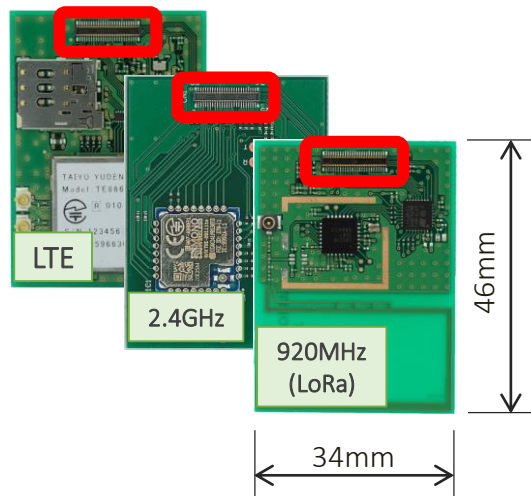
SDカードほぼ2枚分に収まるコンパクトサイズ
モノ側の通信に必要なすべてを凝縮



IoTデバイス/サービスの開発工数を削減
「作らなければならないもの」より「作りたいもの」に注力可能



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



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

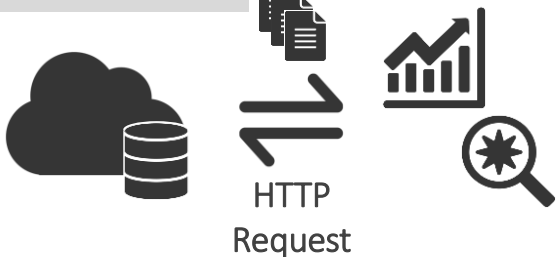
デバイスから送られたデータはポリシーに応じてプラットフォーム内のデータストアに自動で保存される



リアルタイム連携



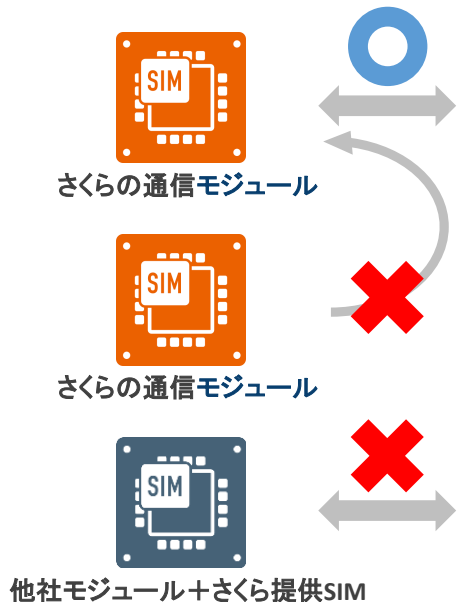
一括連携



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

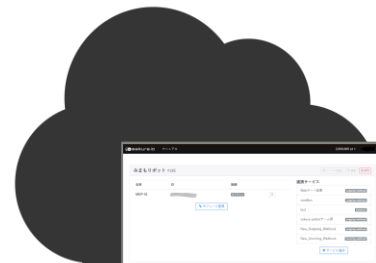
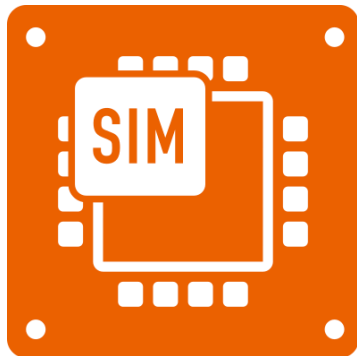


デバイスからプラットフォームまでは閉域網経由、他デバイス含め外部アクセスはAPI経由でのみ行うことでセキュリティを担保

- 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ェアで実装
 - モノ作りの人にTCP/IPやSSLを実装させない



時刻提供機能

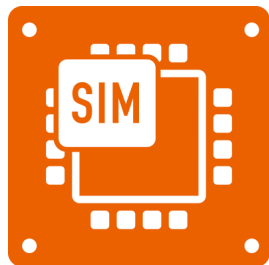


※一部マイコン側での対応が必要です

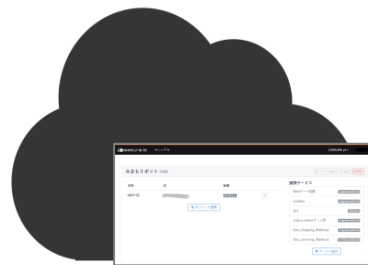
スケジュールでの一斉動作といった場合でも
通信モジュールから時刻情報を提供可能



簡易位置情報提供機能



基地局情報



簡易的な
位置情報

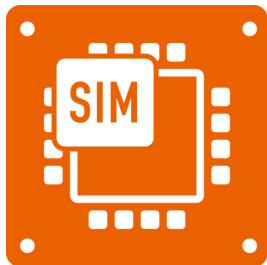


※利用には別途月額料金が必要になります

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



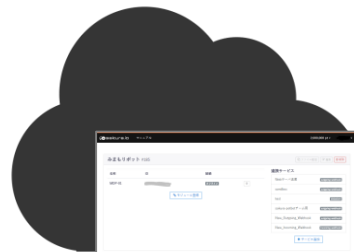
ファイル配信機能



ファイル要求



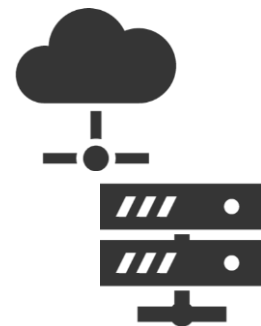
ファイル配信



ファイル要求



ファイル配信



※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

汎用/特定サービスとの連携



世界中で利用できる

Microsoft Azure

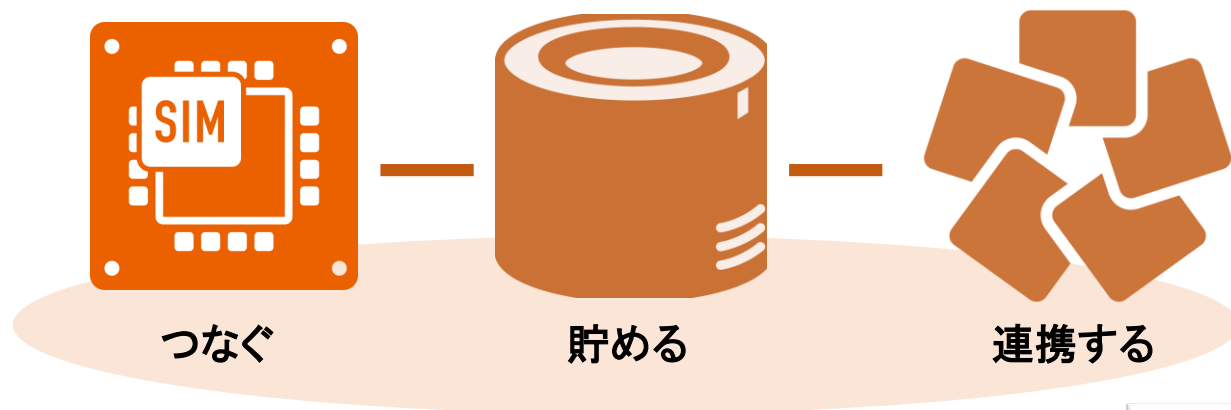


WebSocket



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



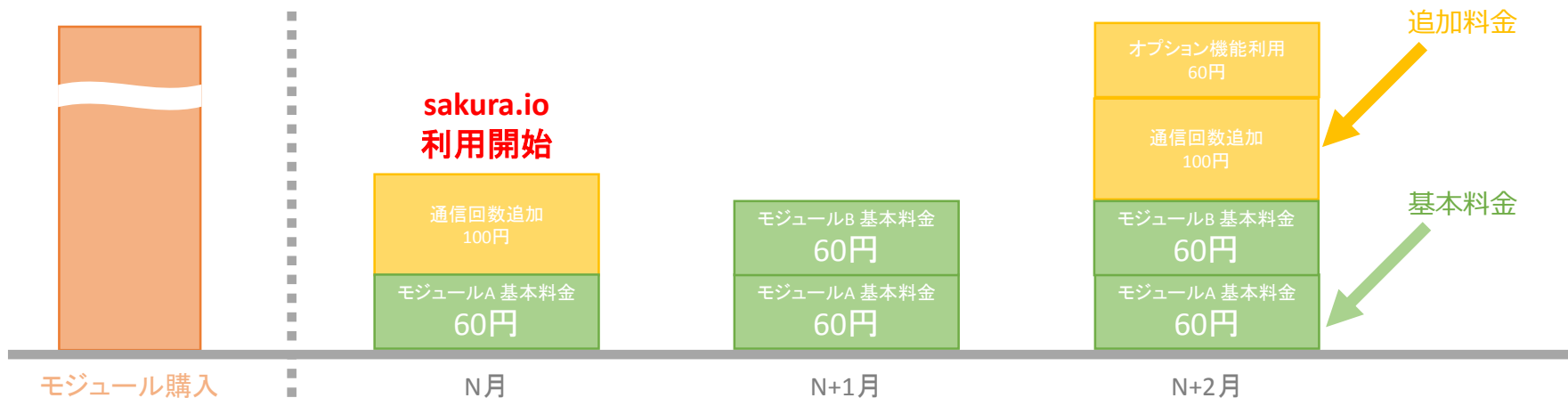
•基本料金

✓ **60円/月** ※毎月1万回分の通信が可能なポイントを付与

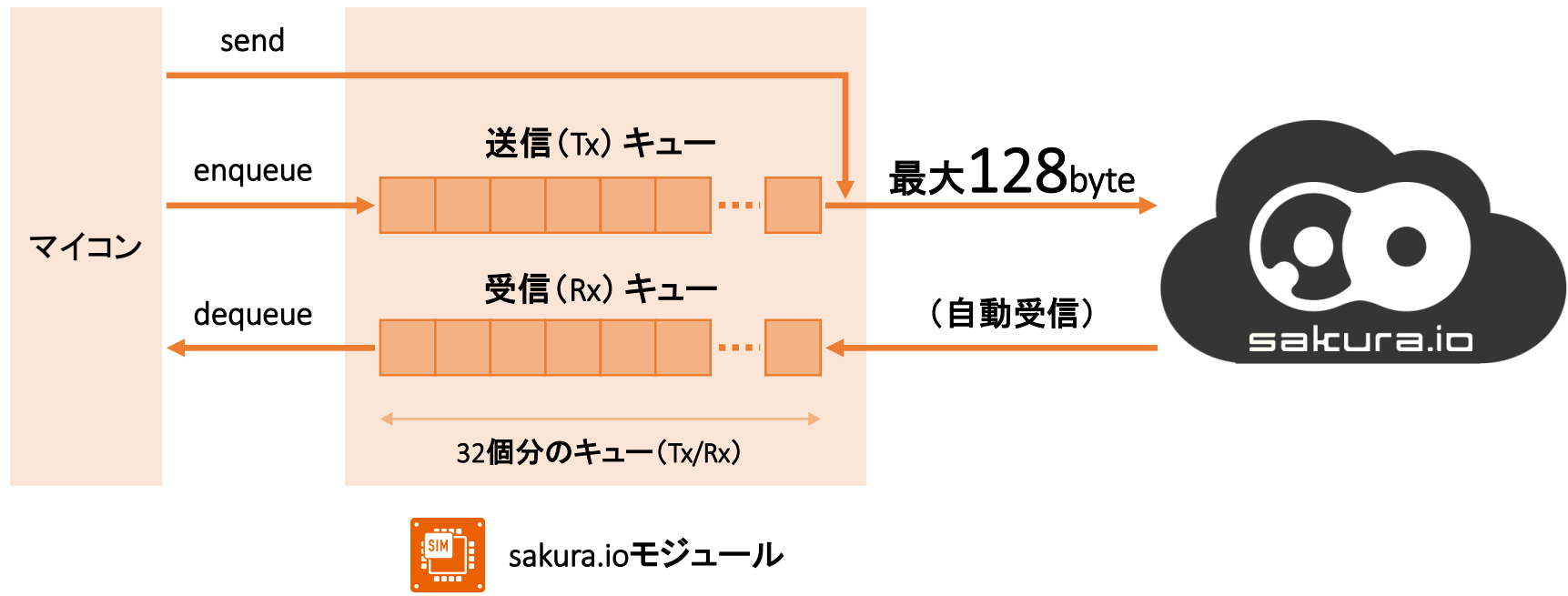
•追加料金

✓ 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



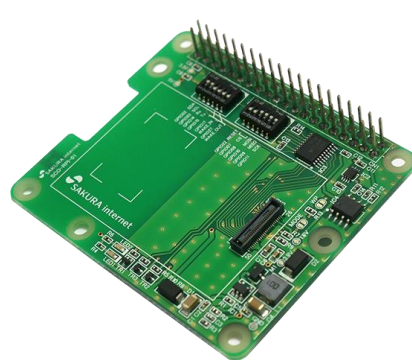
1回の送受信データ(メッセージ)は、最大「128byte (8byte x 16)」
時間や送信元情報はプラットフォーム受信時に付与、マイコン側対応不要



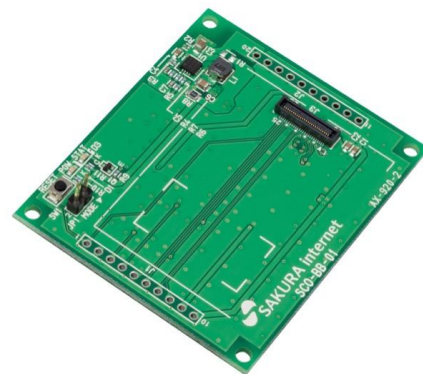
LTE通信モジュール



Arduino用シールド



Raspberry Pi用HAT



ブレイクアウトボード

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 ブレイクアウトボード	オプション	2,500円
sakura.io HAT for Raspberry Pi NEW!!	オプション	5,000円

※モジュール金額は個包装のもので
※すべて税別表記です

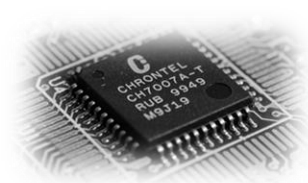
提供方式	数量	販売経路	用途	備考
個包装	1個～	店舗 ECサイト	個人利用 プロトタイプ生産 少量製品	アンテナ同梱
トレイ	90個単位	ECサイト 商社	少～中量生産	アンテナ無し
プロトコル ライセンス (LTE/920MHz)	応相談	お問い合わせ ください	独自設計マイコン への相当機能追加	初期費用 + サポート費用

ライセンス方式での提供で、より柔軟な設計製造が可能に

「自社ソフトウェアに組み入れたい」に対応

通信機能の「つくりかた」を提供

	さくら	お客様
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



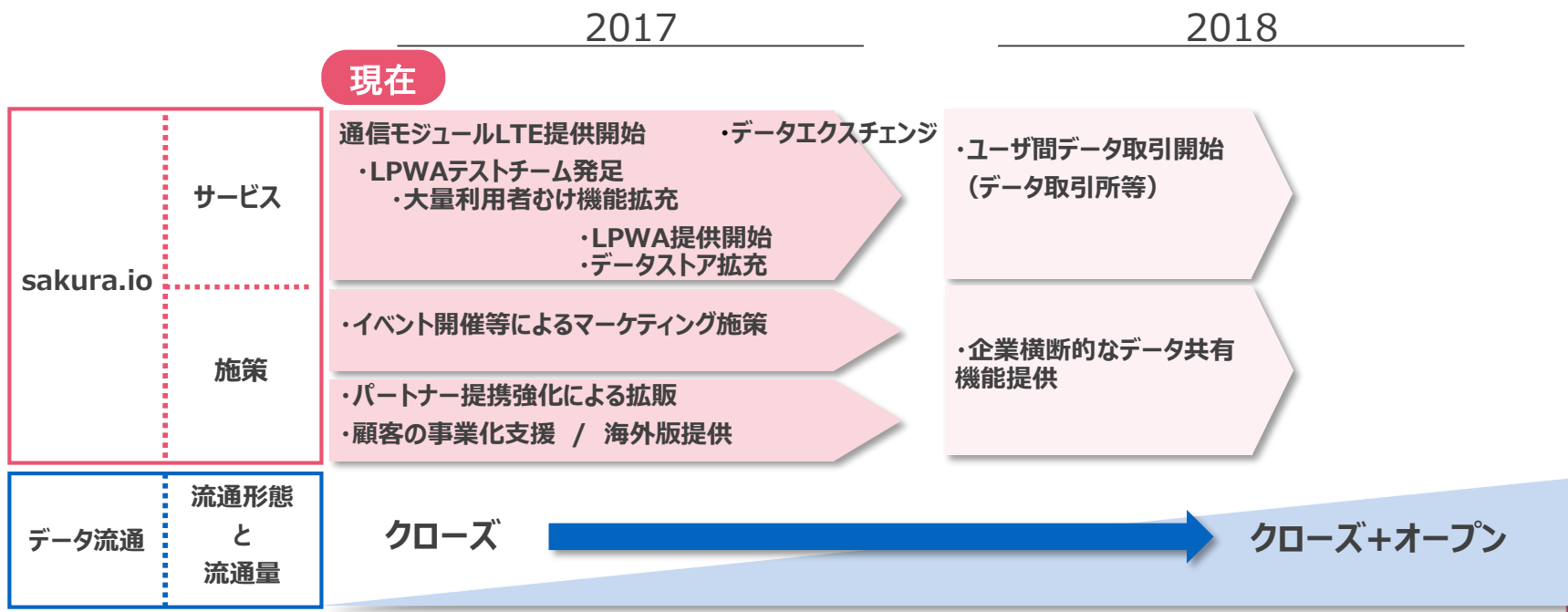
※ ファームウェアアップデートは自社で実施する必要があります。



T!NK <https://tinklock.com/> 株式会社tsumug <https://tsumug.com/>
TechCrunch記事 <http://jp.techcrunch.com/2017/11/09/tsumug-tink/>

アパマンショップホールディングスと連携、2021年までに
賃貸管理物件100万世帯への設置を目指す

新製品、新機能を拡充し、市場拡大を目指す



新規 製造製品向け



プラットフォームの提供

SIM付き通信モジュールをコアとした
IoTプラットフォームの提供



単体方式
LTE



ゲートウェイ方式
LoRa



接続方法開示
+
SIM供給

IoTに必要なプラットフォーム機能を提供

つなぐ

保存する

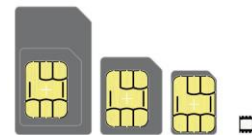
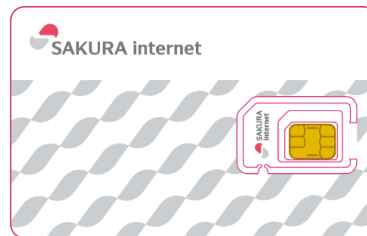
連携する

既存 製品にも NEW



IoT向けSIMの提供

クラウドにダイレクトに接続し、任意のネットワークへ
接続可能なSIM。セキュリティを担保し、
国内で最も安価な通信を提供



通常SIM ~ MFF2まで

IoTに求められる通信機能を提供

安全

安価

高速通信

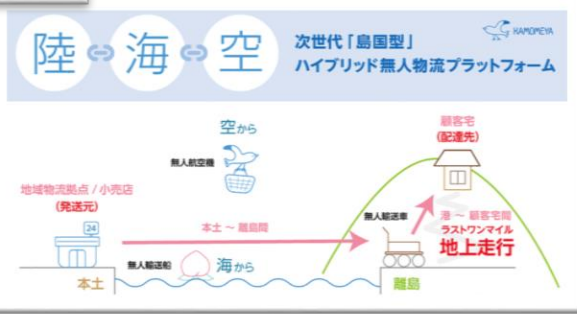
電気信号とJSONを相互変換する プラットフォーム



これまでのモノ作り/サービス作りを補完
現在持っている技術/事業範囲で共創が可能

利用事例

公共 教育 農業 健康 新分野 製造



- イノシシやマンゲースの駆除に利用
- 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

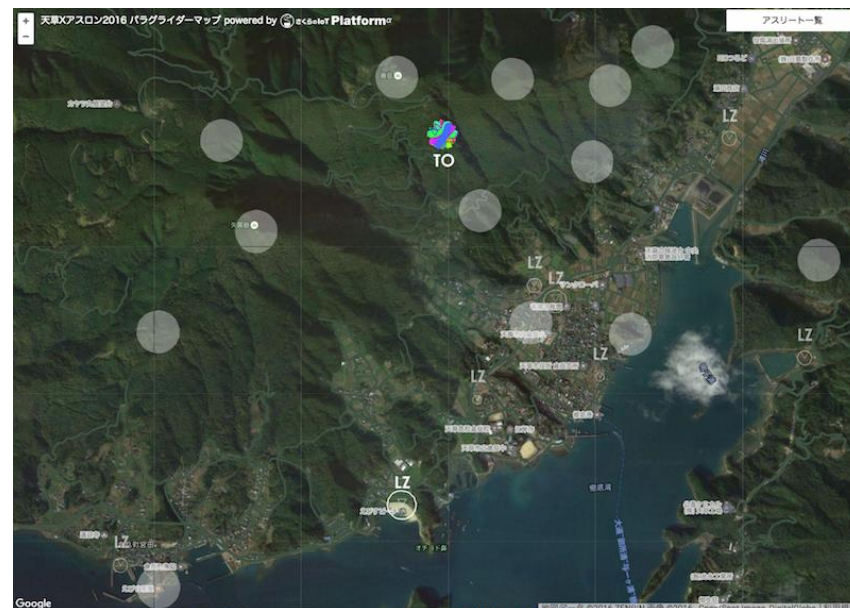


https://www.sakura.ad.jp/press/2017/0330_sabae-tsutsujibus/

- 制作：ゼロスペック株式会社
- 灯油タンクにセンサーと通信モジュールを設置
- 灯油の残量を計測し送信
- 残量が少なくなったら配送
- <https://www.zero-spec.com/>



- 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>



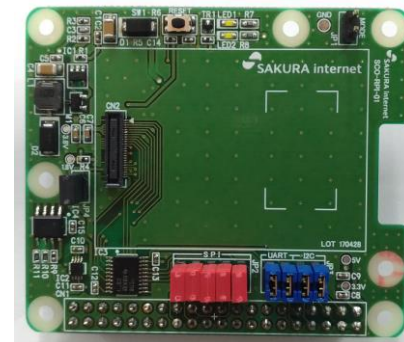
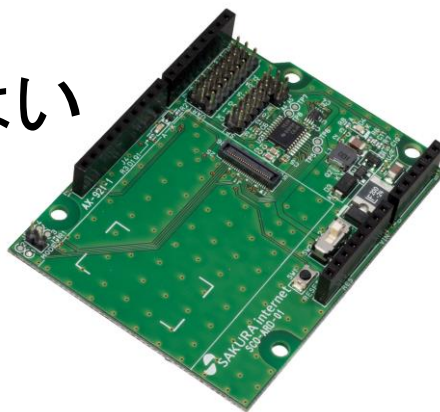
sakura.ioと
OSSの組み合わせ

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
- 接続用ボードを販売
 - Arduino用シールド
 - Raspberry Pi用HAT



- Arduino用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/sakura-internet/SakuraAlphaArduino>

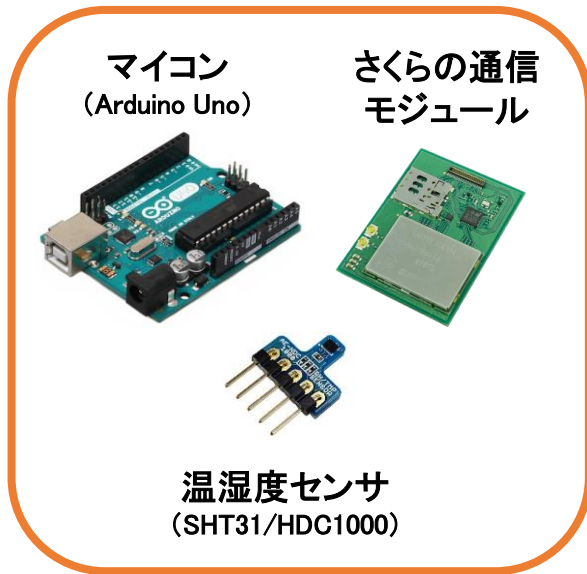
- sakura.ioで用意したAPIによりリクエストを送出
- 対応プロトコル/サービス
 - WebSocket
 - Webhook (Incoming / Outgoing)
 - MQTT
 - DataStore API
 - AWS IoT
 - Azure IoT Hub
- URLの例
 - wss://api.sakura.io/ws/v1/(ID)
 - https://api.sakura.io/incoming/v1/(ID)
- JSON形式のデータが返ってくる
- お好みのプログラミング言語でJSONデータを処理
 - 主要な言語はJSONを扱うライブラリあり

```
{
  "module": "XXXXXXXXXX",
  "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)

温湿度データを取得し
sakura.ioへ送付

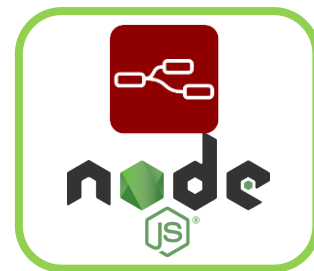
Websocketでデータを入力し
Node-REDで処理



LTE
閉域網



WebSocket



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

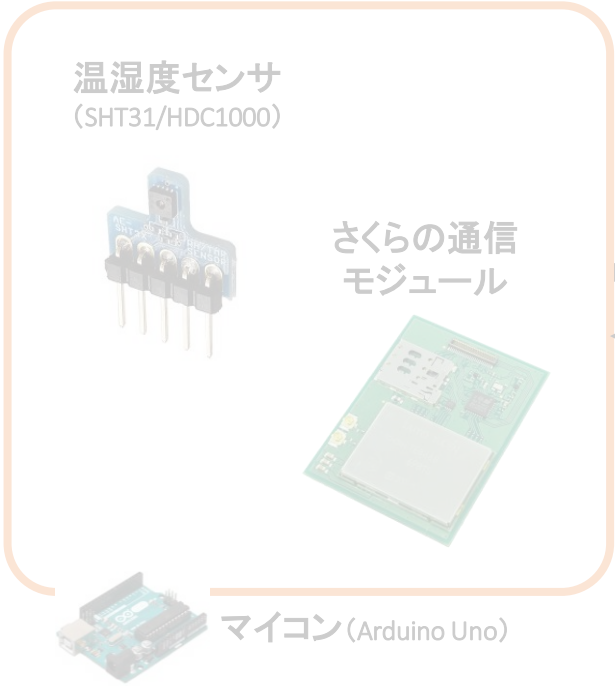
- 通信モジュールとArduinoシールドを接続
- 温湿度センサーからの出力をArduinoに取り込めるよう配線
- 温湿度情報をsakura.ioに出力するArduinoのプログラムを作成

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



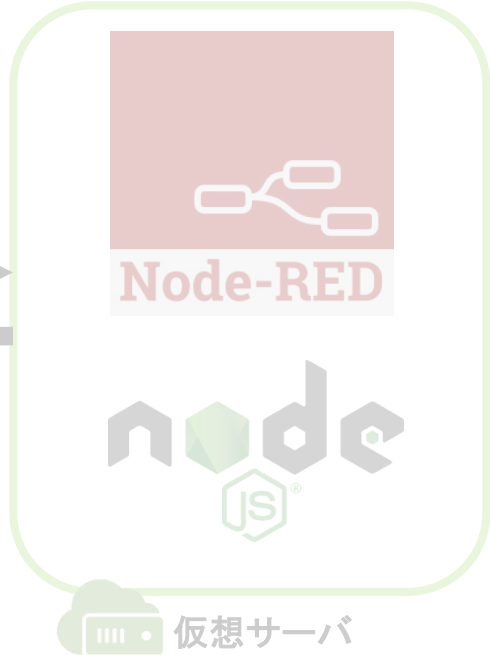
①

sakura.ioの設定



③

Webサービス連携
(さくらのクラウド)



プロジェクトの登録

New Project #2321

ファイル配信

編集

削除

名称	ID	接続
New Module	[REDACTED]	オンライン

モジュール登録

連携サービス

New Service

websocket

+ サービス追加

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。

追加サービスの選択

WebSocket

Outgoing Webhook

Incoming Webhook

MQTT Client

DataStore API

AWS IoT

Azure IoT Hub

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。

サービス連携の編集 WebSocket #3656

名前

New Service

URL

wss://api.sakura.io/ws/v1/

Token

削除

保存

②

マイコンおよびプログラムの構築

温湿度センサ
(SHT31/HDC1000)



さくらの通信
モジュール



マイコン (Arduino Uno)

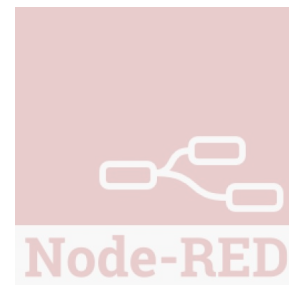
①

sakura.ioの設定



③

Webサービス連携
(さくらのクラウド)



仮想サーバ

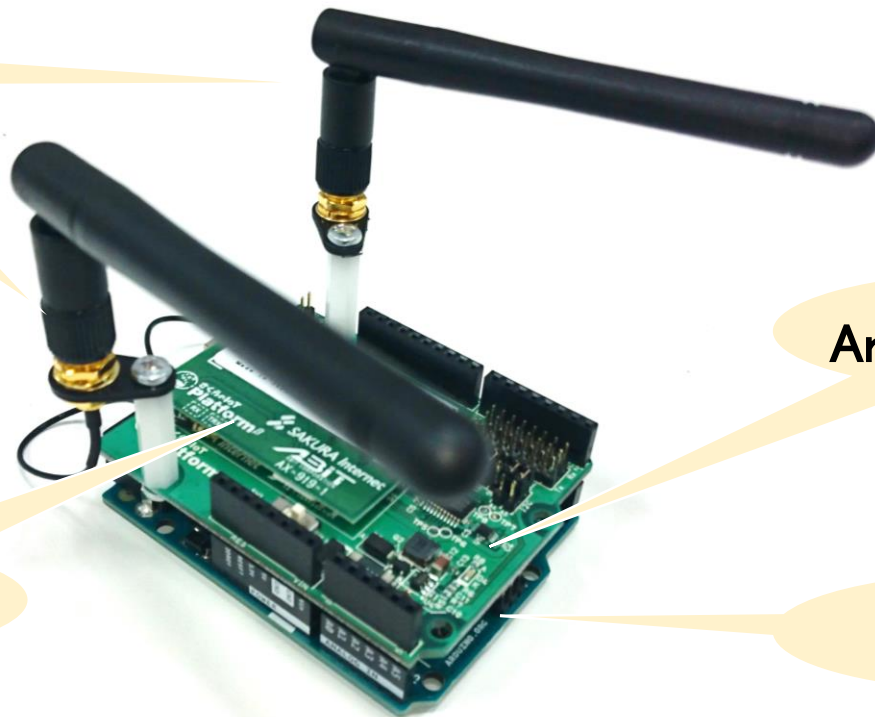
通信モジュールの下にArduinoシールドを敷き、さらにその下にArduinoを接続します。
通信モジュールにはLTEアンテナを取り付けます。

LTEアンテナ

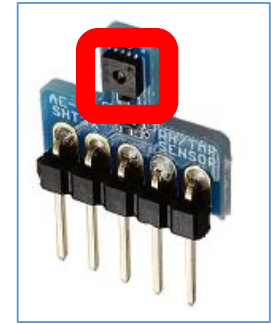
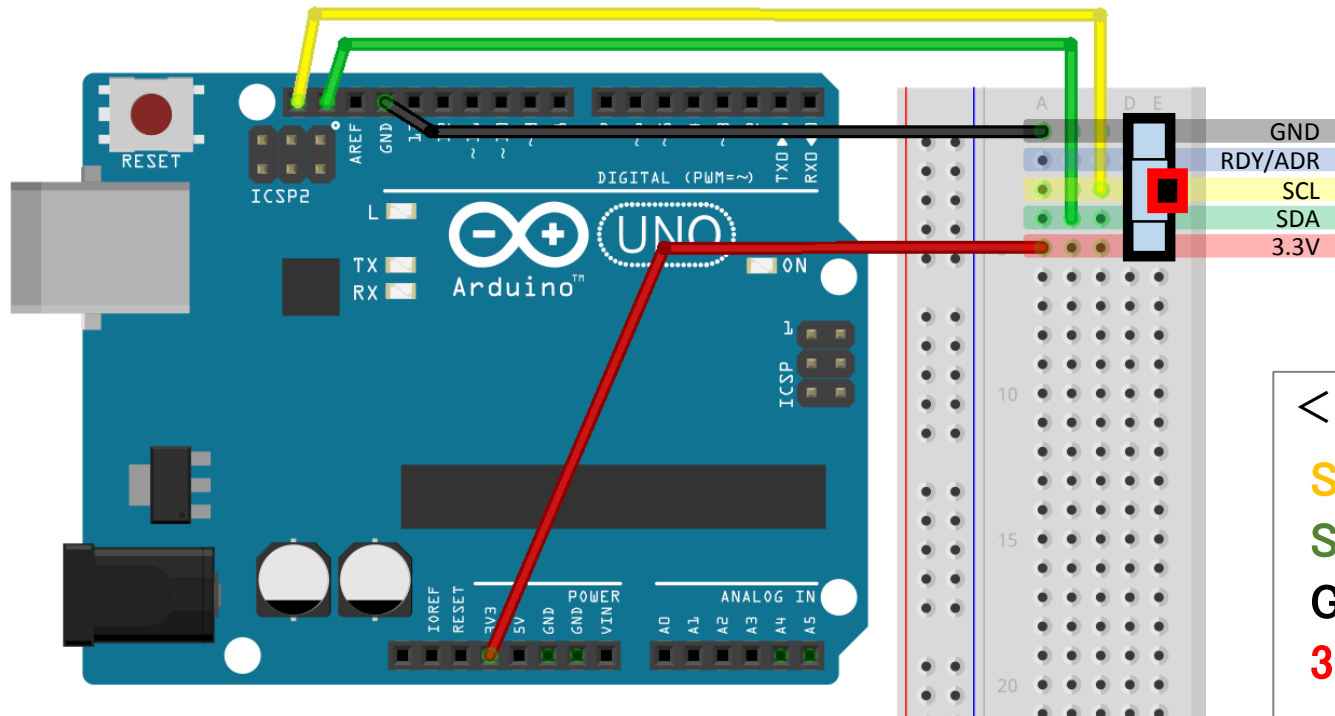
Arduinoシールド

通信モジュール

Arduino



温湿度センサーをブレッドボードに設置し、図のように配線します。
(実際にはArduinoシールドに対して配線します)
SCL, SDAの2本の線でI2Cによる通信を行います。



<凡例>

SCL	—
SDA	—
GND	—
3.3V	—

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。
(プログラムはGitHubで公開しています。https://github.com/sakuraio)
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。

```
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)
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
```

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。

時刻:
データがモジュールのキューに格納された時刻のタイムスタンプ

モジュール:
データを送信した通信モジュールのID

チャンネル:
データが格納されたチャンネル番号

型:
データの型式

値:
送信された値

詳細モードに切り替え

時刻	モジュール	チャンネル	型	値
2017-09-19T03:32:35.760758677Z	[REDACTED]	0	f	27.883957
2017-09-19T03:32:35.782758677Z	[REDACTED]	1	f	43.07469
2017-09-19T03:32:35.804758677Z	[REDACTED]	2	l	21889
2017-09-19T03:32:03.852397653Z	[REDACTED]	0	f	27.91066
2017-09-19T03:32:03.874397653Z	[REDACTED]	1	f	42.899216
2017-09-19T03:32:03.896397653Z	[REDACTED]	2	l	21888

→ 温度

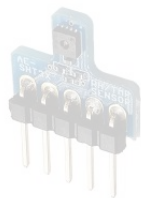
→ 湿度

→ カウント値

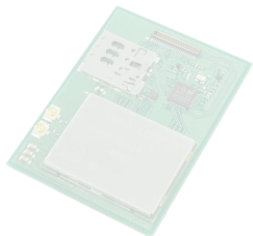
②

マイコンおよび
プログラムの構築

温湿度センサ
(SHT31/HDC1000)



さくらの通信
モジュール



マイコン (Arduino Uno)

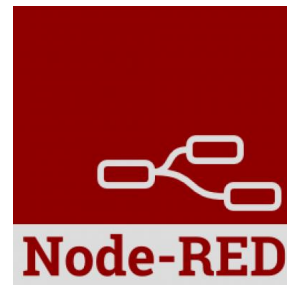
①

sakura.ioの設定



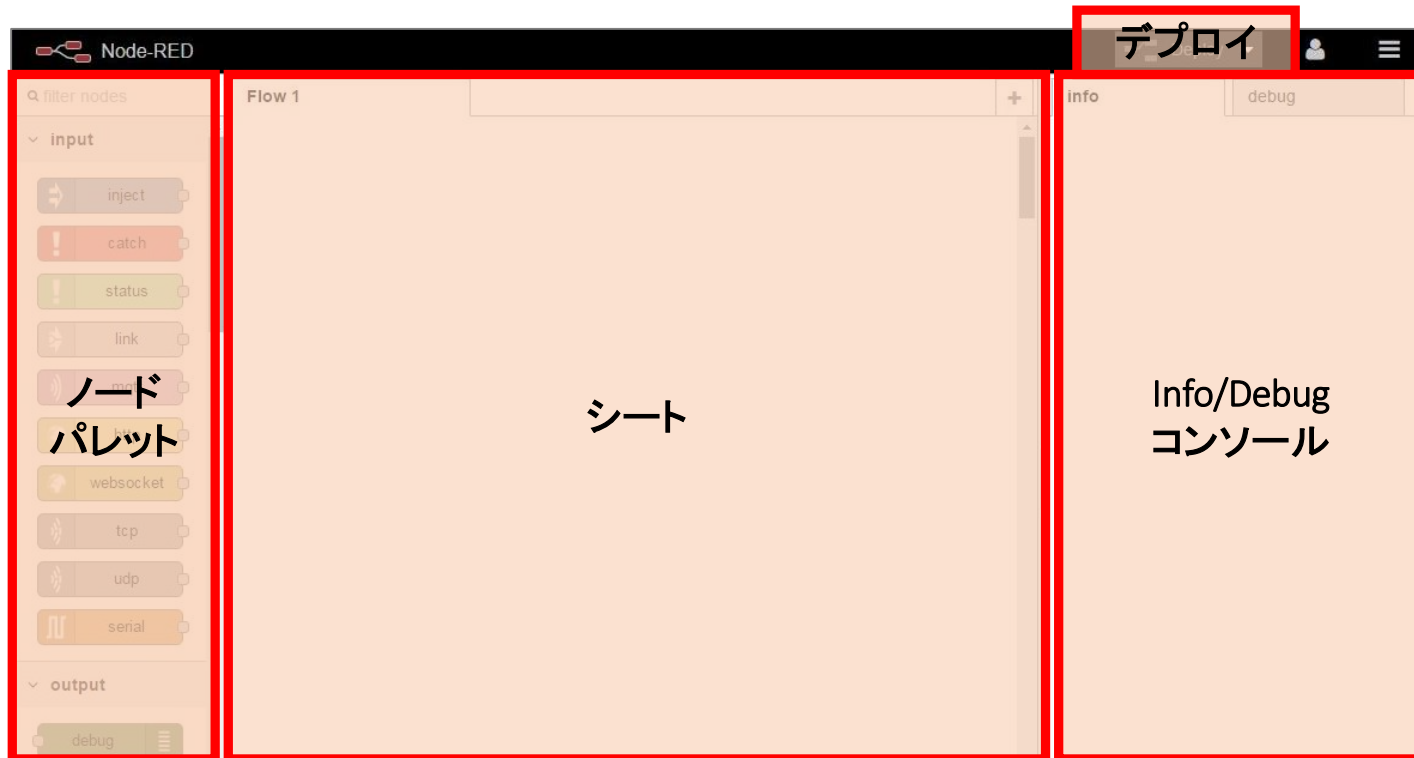
③

Webサービス連携
(さくらのクラウド)



仮想サーバ

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。



スタートアップスクリプト

なし shell yaml_cloud_config

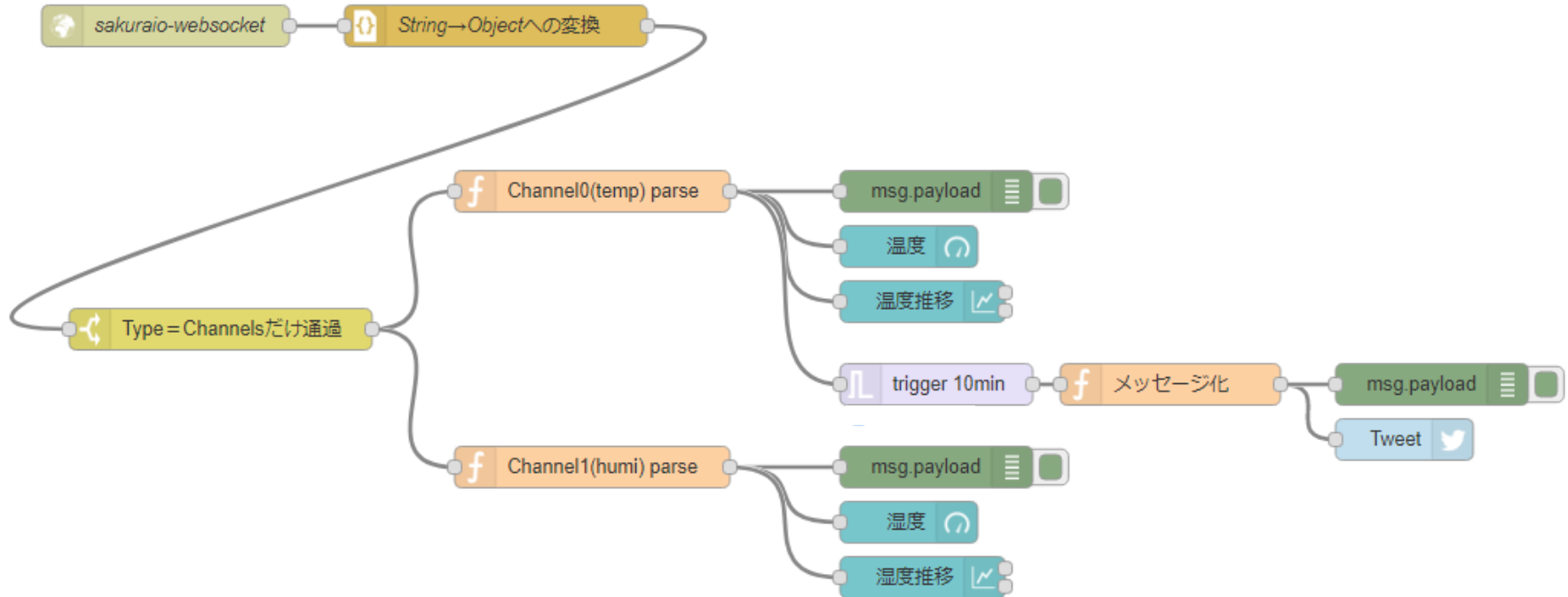
詳細は[技術仕様](#)をご確認ください

配置する スタートアップスクリプト

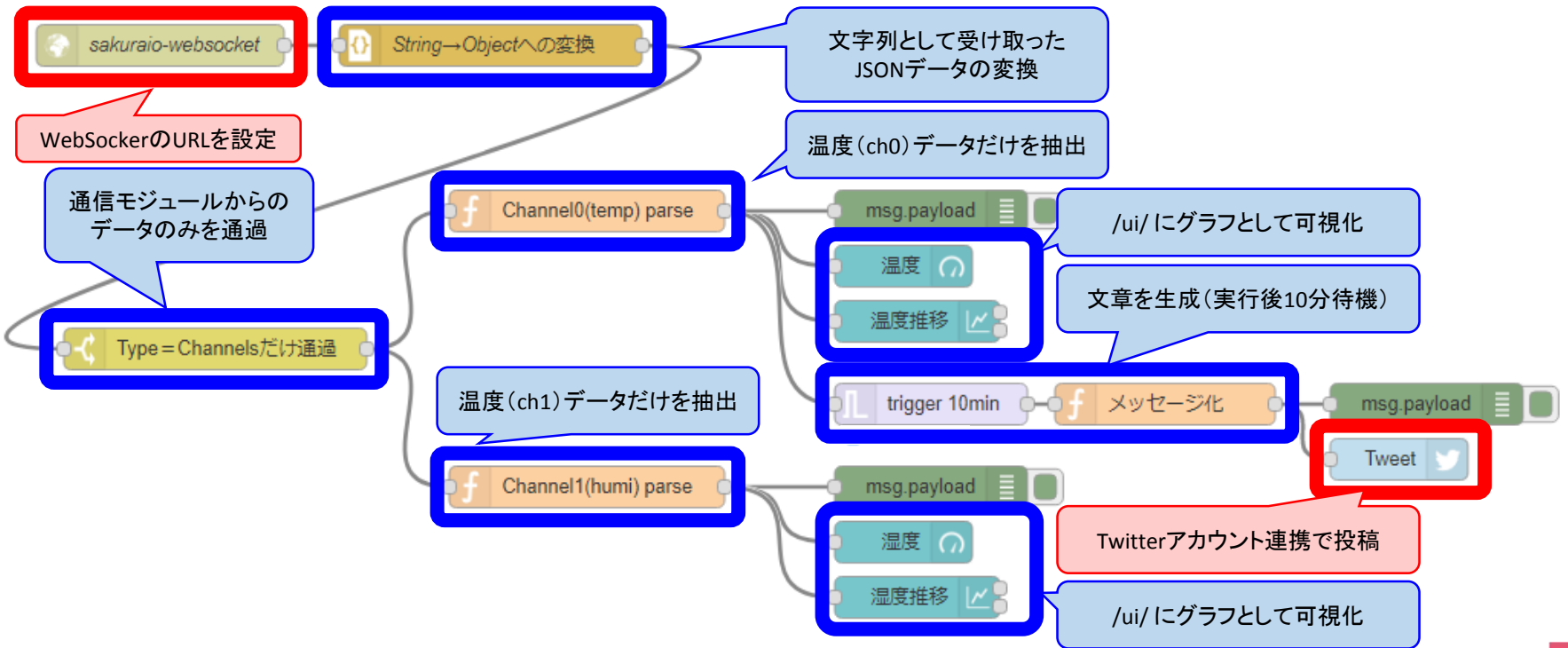
[public] Node-RED #112900523333

• NVM/Node.js/Node-REDのインストールを実行します。
このスクリプトは、CentOS 7.xでのみ動作します。
完了後「http://<IPアドレス>:1880/」にWebブラウザからアクセスできます。
UIポート番号を指定した場合は、指定したポート番号でアクセスできます。
Node-Redのログを確認するには「pm2 logs node-red」コマンドを実行します。

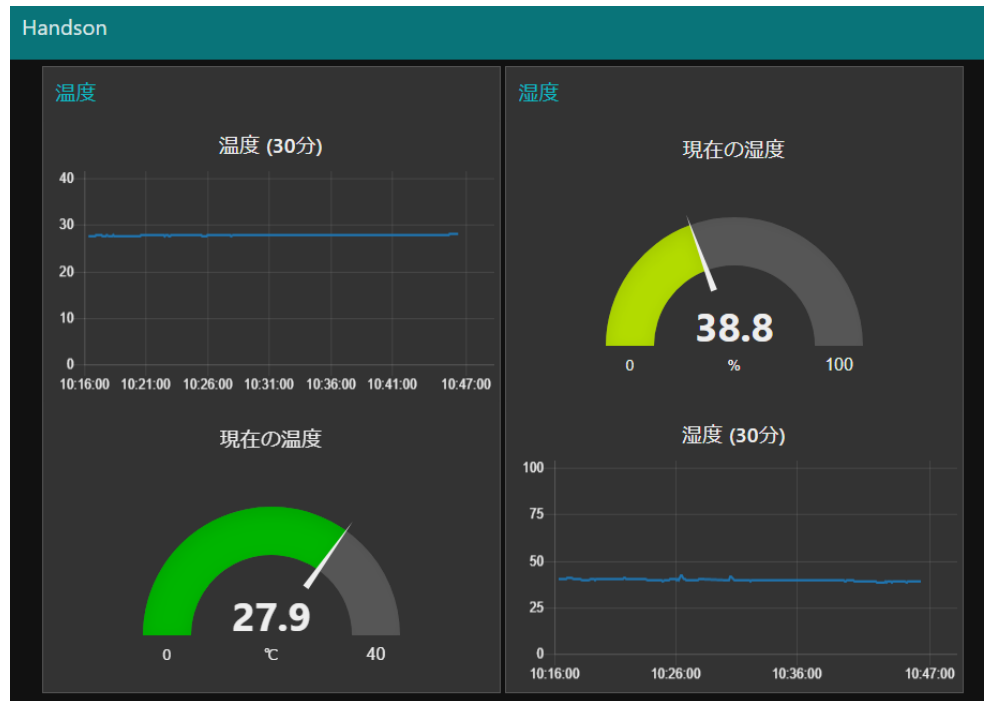
sakura.ioからWebSocketで温湿度データを手し、加工して表示するフローを作成します。
WebSocketのURLとTwitterのIDを設定し、デプロイすると動作します。



フローの内部ではこのような処理をしています。



【 <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。



フローをデプロイすると、その時取得した温度センサの情報が文中に埋め込まれた状態で、設定したTwitterアカウントにツイートが投稿されます。

[メッセージ化] ノードの内容を書き換えることで任意の文章に変更することができます。



██████████ 15 秒

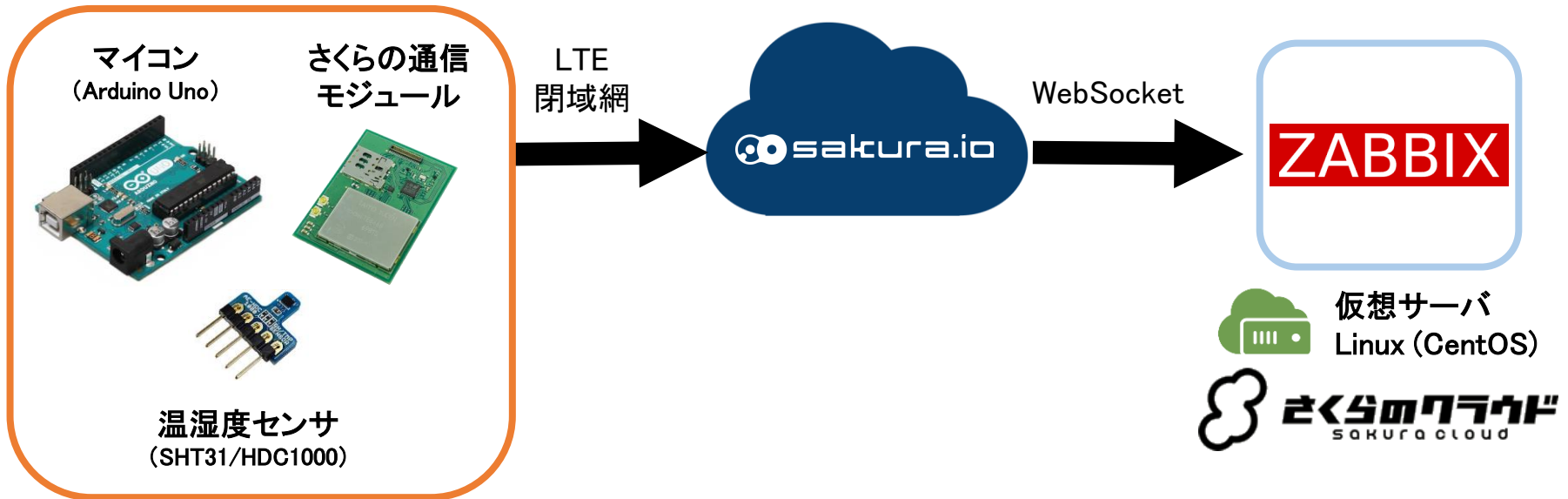
さくらインターネットのハンズオンで温度情報を取得中！ただ今の現地温度は27.8度だよ。LPWAを使用したゲートウェイ方式も鋭意提供準備中です！今後ともよろしくお願いします！ #sakuraio #さくらインターネット



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へ送付

Websocketでデータ入手し
Zabbixに投入



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

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サーバを簡単に作成することができます。



スタートアップスクリプト

なし shell yaml_cloud_config

詳細は[技術仕様](#)をご確認ください

配置する スタートアップスクリプト

• このスクリプトはZabbix Serverをセットアップします。(このスクリプトは、ZabbixのURLは http://IP Address/zabbix です。)

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

The screenshot shows the Zabbix web interface. At the top, there is a navigation bar with the ZABBIX logo and menu items: 監視データ, インベントリ, レポート, 設定, 管理. Below this is a secondary navigation bar with: ホストグループ, テンプレート, ホスト, メンテナンス, アクション, イベント, 相関関係, ディスカバリ. The main heading is 'アイテム'. Below the heading, there is a breadcrumb trail: 'すべてのホスト / Zabbix server' followed by a '有効' status indicator and several filters: ZBX, SNMP, JMX, IPMI. To the right, it shows 'アプリケーション 12' and 'アイテム 73'. The main content area shows the configuration for a new item, with a red box highlighting the following fields:

名前	Temperature
タイプ	Zabbix-ラッパー
キー	sakura_iot_temp
データ型	数値 (浮動小数)
単位	°C

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";
```

テスト WebSocket

名前

Token

████████████████████

WebSocket

wss://api.sakura.io/ws/v1/████████████████████

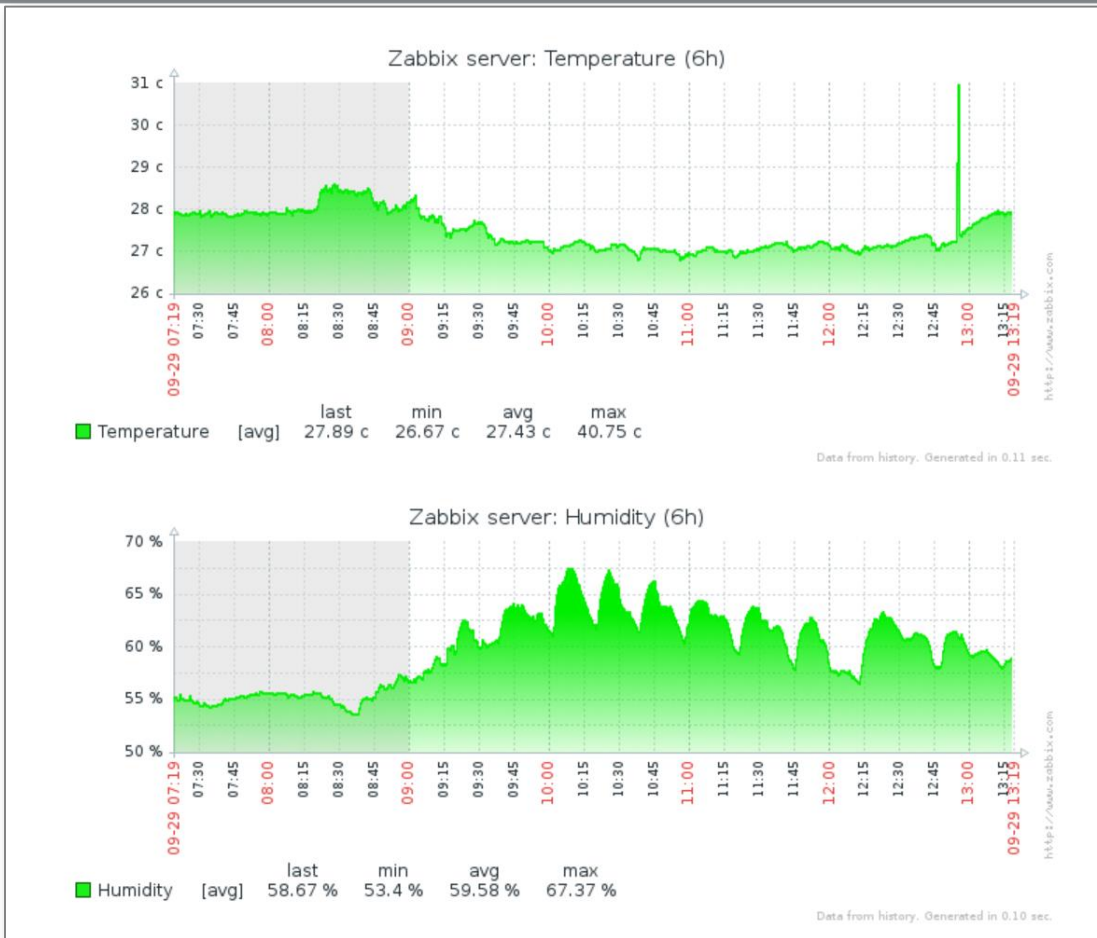
受信データ

時刻	モジュール	チャンネル	型	値
----	-------	-------	---	---

< 戻る

削除する

+ 保存する



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



さくらの通信
モジュール

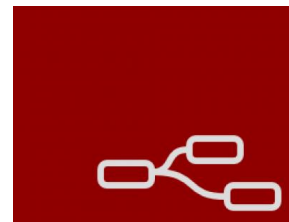


マイコン (Arduino Uno)

LTE
閉域網



WebSocket



Node-RED



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

②

マイコンおよび
プログラムの構築

抵抗入りLED



さくらの通信
モジュール



マイコン (Arduino Uno)

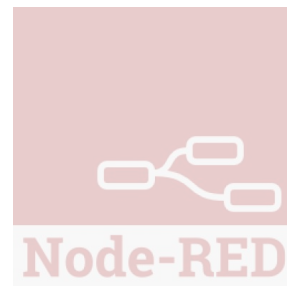
①

sakura.ioの設定



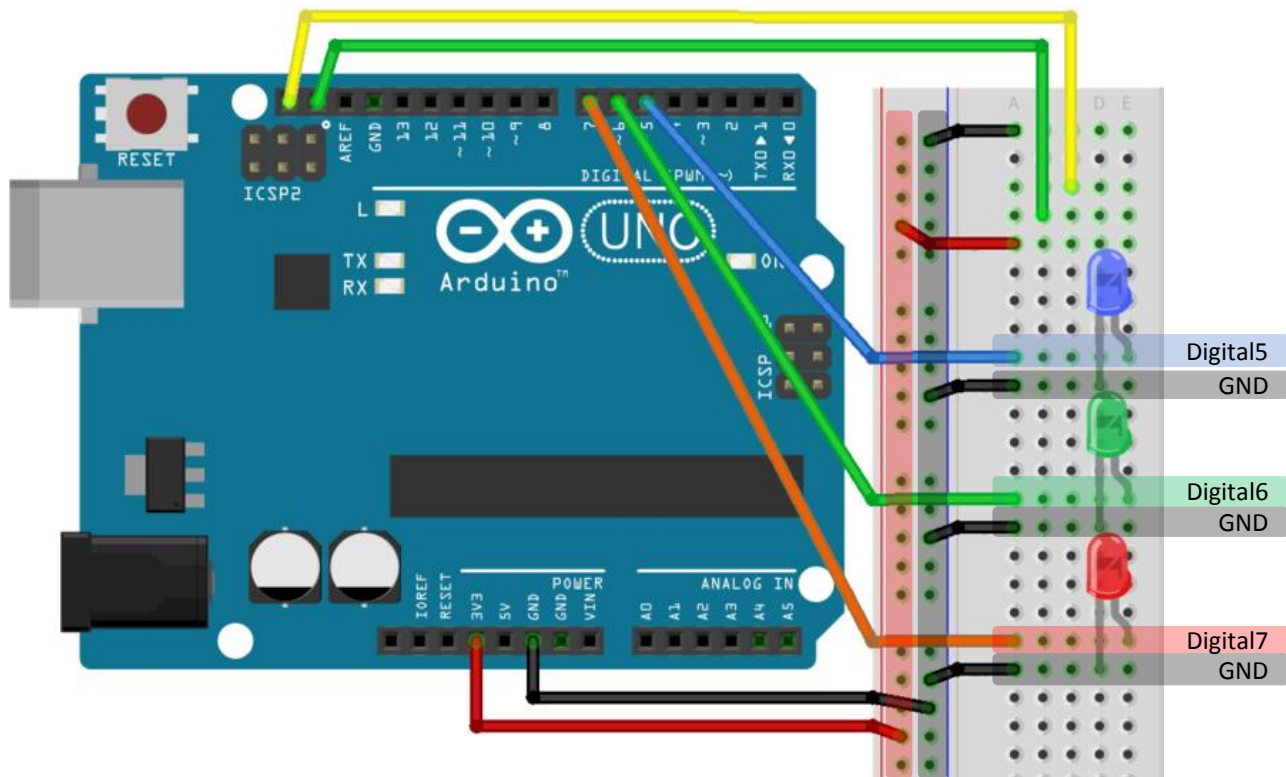
③

Webサービス連携
(さくらのクラウド)








仮想サーバ

LEDをブレッドボードに設置し、図のように配線します。
(実際にはArduinoシールドに対して配線します)



<凡例>

- D5 
- D6 
- D7 
- GND 
- 3.3V 

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。
(プログラムはGitHubで公開しています。https://github.com/sakuraio)
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;
9
```



```
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
```

②

マイコンおよび
プログラムの構築



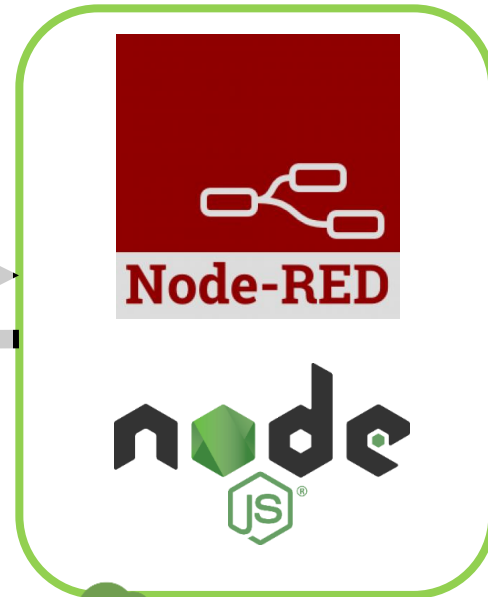
①

sakura.ioの設定

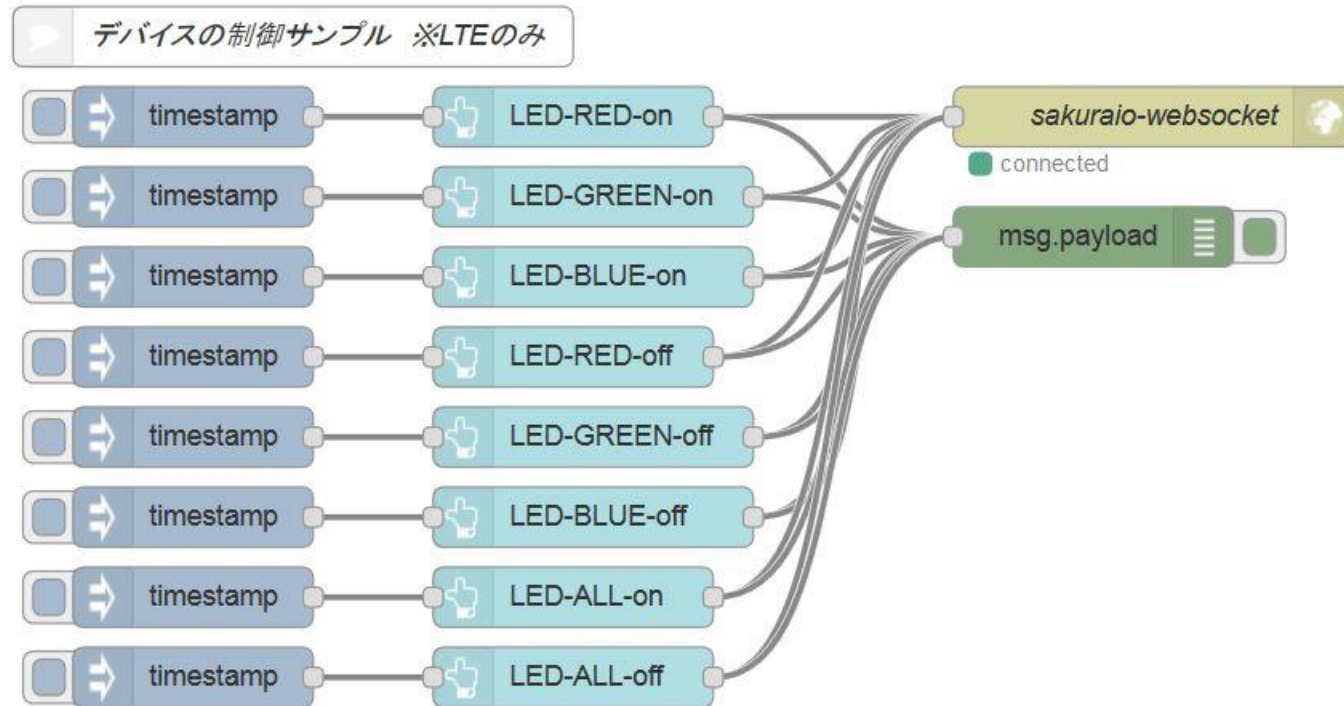


③

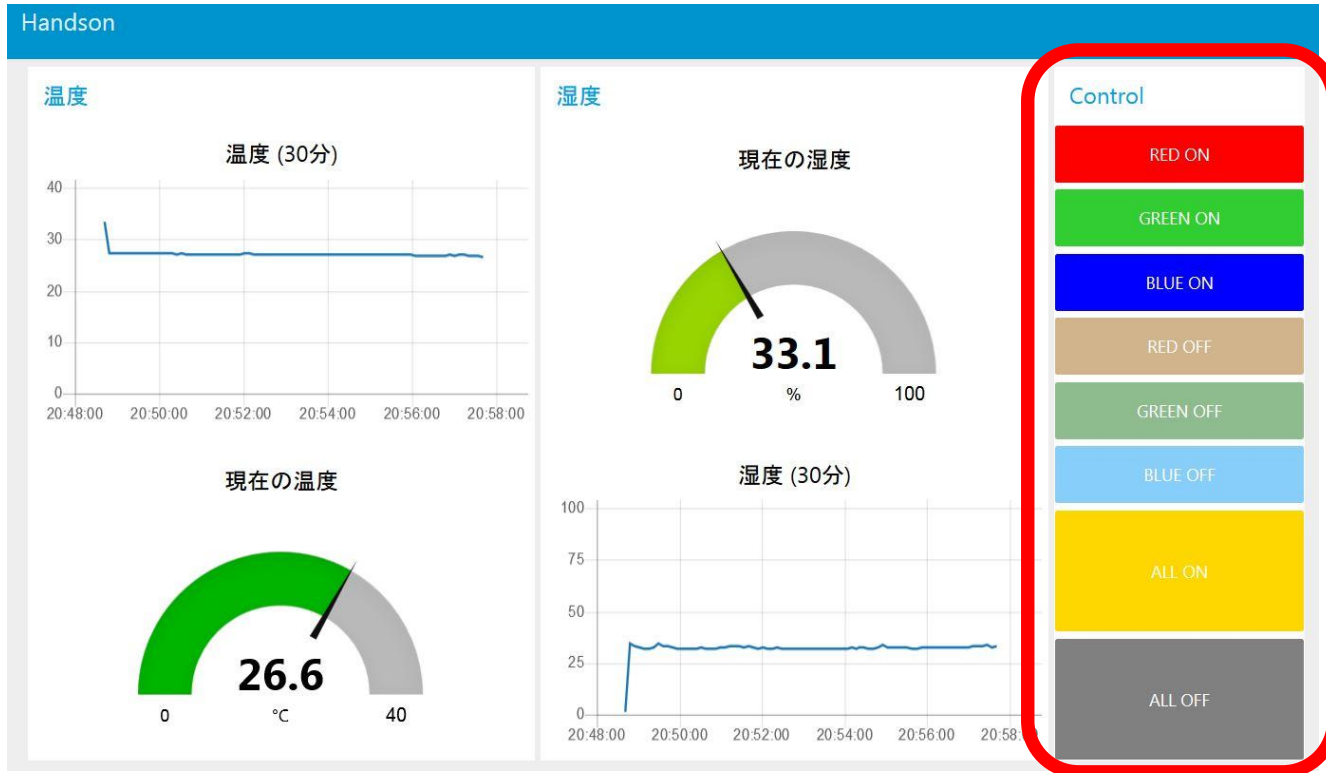
Webサービス連携
(さくらのクラウド)



左端のボタンをクリックすると赤/緑/青のLEDをON/OFFするフローを作成します。
WebSocketのURLと、LEDノードに通信モジュールのIDを設定し、デプロイすると動作します。



【 <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

- 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



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

- 今日の講演で紹介した内容を実習するハンズオン
- 東京では毎月開催中
- パートナーとの共催ハンズオンも実施中
 - 駅すばあと / myThings / Azure / AWS / Bluemix / Twilio など
- 開催予定: 2/26(月)東京、3/15(金)福岡(Twilio)、4/14(土)広島
- イベントページ
 - さくらのイベント(関東版): <https://sakura-kanto.doorkeeper.jp/>
 - さくらのイベント(九州版): <https://sakura-kyushu.doorkeeper.jp/>
 - さくらのイベント(中四国版): <https://sakura-chushikoku.doorkeeper.jp/>



- sakura.ioを使ったIoT工作の解説書
- 2018年2月2日発売 **[New!]**
- セミナーで紹介した内容はほぼ掲載
- それ以外の内容
 - 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)
- 展示ブースに書籍あり

- 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

参加者アンケートにご協力ください



ご回答いただいた方に粗品を進呈します