

# sakura.io 体験ハンズオン

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

DAY

2017/9/2

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 (元会長)
  - 全国のOSCなどで研究会を開催
  - 多種多様なコミュニティとイベントを共同開催
- ・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)
  - 技術記事の執筆
- ・くわしくは「法林浩之」で検索

スライドシェア

# SlideShareのさくらインターネットアカウント



[http://www.slideshare.net/sakura\\_pr/](http://www.slideshare.net/sakura_pr/)

## ウェブアクセラレータ紹介資料

SAKURA Internet さくらのクラウド

[No.1 概要] [No.2 機能と仕様] [No.3 設定手順] [No.4 料金] [No.5 リアルンス]

# さくらのクラウド ウェブアクセラレータ

Ver1.1 (2016年10月3日版)

正式サービス版

DATE	COMPANY	SERVICE	VERSION
2016/10/3	さくらインターネット株式会社	さくらのクラウド	1.1

## シンプル監視アプライアンス導入ガイド

SAKURA Internet さくらのクラウド

[No.1 概要] [No.2 機能と仕様] [No.3 設定手順] [No.4 料金] [No.5 リアルンス]

# さくらのクラウド シンプル監視

Ver1.00 (2016年8月26日版)

DATE	COMPANY	SERVICE	VERSION
2016/8/26	さくらインターネット株式会社	さくらのクラウド	1.00

はじめに



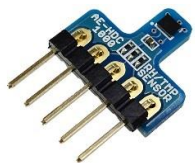
1. 本ハンズオンはsakura.ioを使用し、組み込み系エンジニアおよびWeb/アプリ開発系のエンジニアがご自身のスキルセットを大きく超えることなく、Internet of Things(IoT)に挑戦できることを体験いただくものです。
2. そのため各章内で技術的な詳細は極力省略しております。
3. 今回は1人1つワークショップキットをご用意しておりますが、組み込みやWeb/アプリ開発に詳しい方がいらっしゃいましたら、ご不明点を積極的にフォローしあって進めていただければと思います。



#sakuraio

# 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

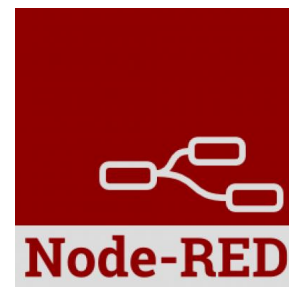


マイコン (Arduino Uno)

# sakura.ioの設定



# Webサービス連携 (さくらのクラウド)



仮想サーバ



## 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. 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)

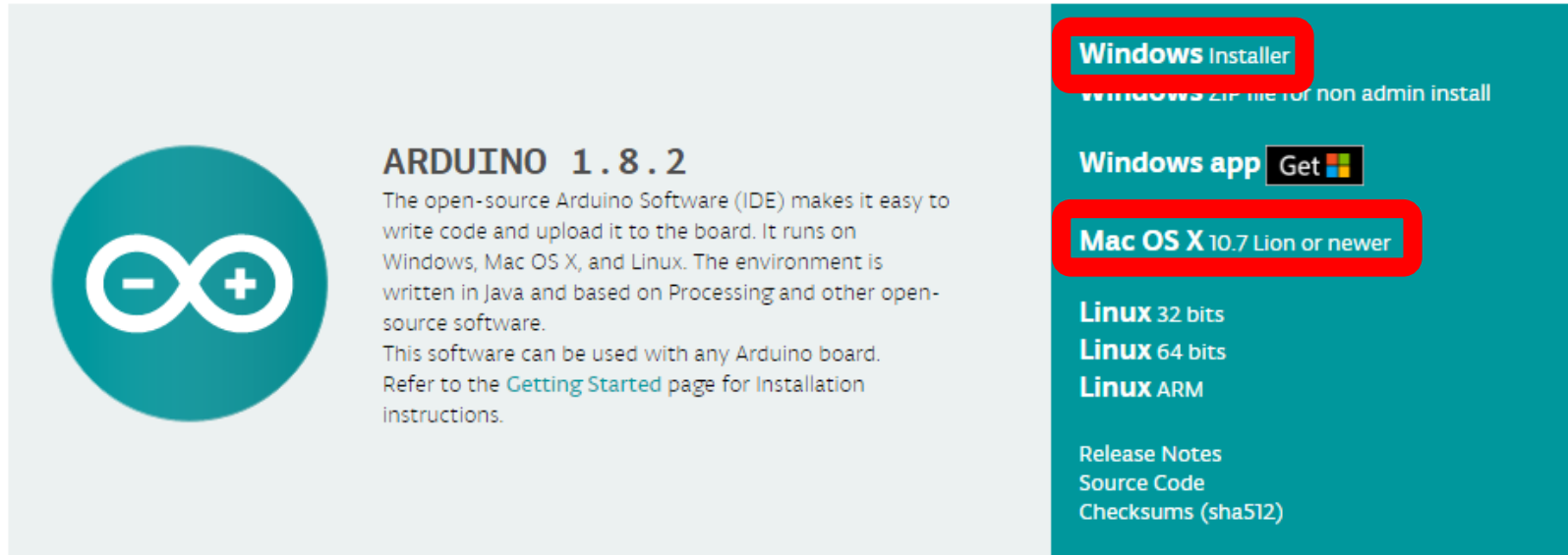
- Node-RED サーバ用インスタンスの作成
- WebSocket を利用したデータ連携フロー作成



# Arduino IDEの セットアップ

https://www.arduino.cc/en/Main/Software から開発環境(Arduino IDE)を入手します。  
2017/8/25時点での最新版は【1.8.4】となります。  
Windowsは【Windows Installer】、Macは【Mac OS X 10.7 Lion or newer】を選択します。

## Download the Arduino IDE



**ARDUINO 1.8.2**

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.

This software can be used with any Arduino board. Refer to the [Getting Started](#) page for Installation instructions.

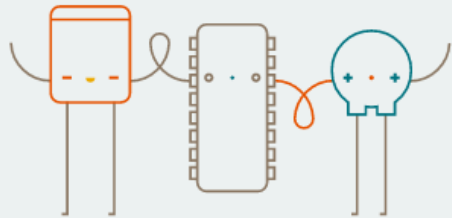
- Windows Installer**
- Windows ZIP file for non admin install
- Windows app [Get](#)
- Mac OS X 10.7 Lion or newer**
- Linux 32 bits
- Linux 64 bits
- Linux ARM

[Release Notes](#)  
[Source Code](#)  
[Checksums \(sha512\)](#)

該当の金額を選択するか(寄付する場合)、もしくは【JUST DOWNLOAD】にてダウンロードします。

## Support the Arduino Software

Consider supporting the Arduino Software by contributing to its development. (US tax payers, please note this contribution is not tax deductible). [Learn more on how your contribution will be used.](#)



SINCE MARCH 2015, THE ARDUINO IDE HAS BEEN DOWNLOADED **14,982,535** TIMES. (IMPRESSIVE!) NO LONGER JUST FOR ARDUINO AND GENUINO BOARDS, HUNDREDS OF COMPANIES AROUND THE WORLD ARE USING THE IDE TO PROGRAM THEIR DEVICES, INCLUDING COMPATIBLES, CLONES, AND EVEN COUNTERFEITS. HELP ACCELERATE ITS DEVELOPMENT WITH A SMALL CONTRIBUTION! REMEMBER: OPEN SOURCE IS LOVE!

\$3

\$5

\$10

\$25

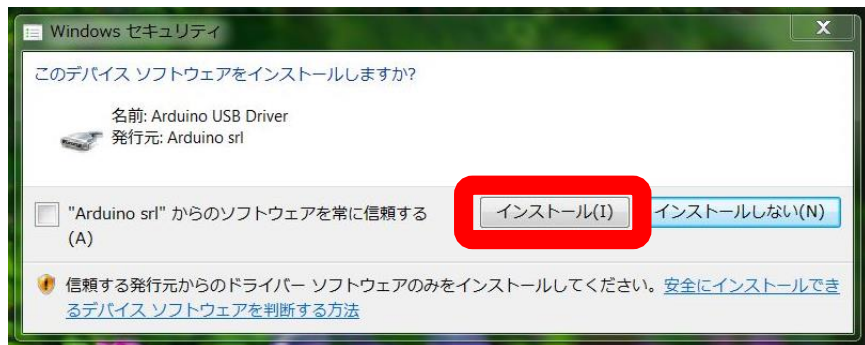
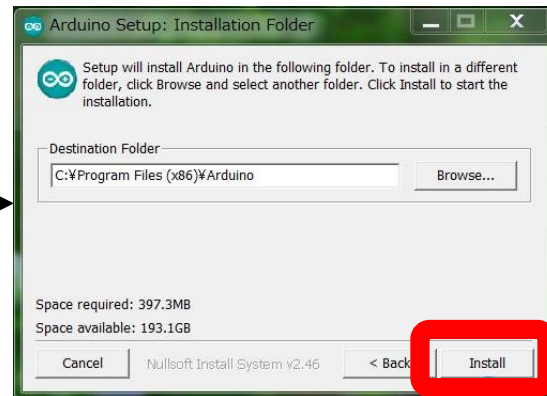
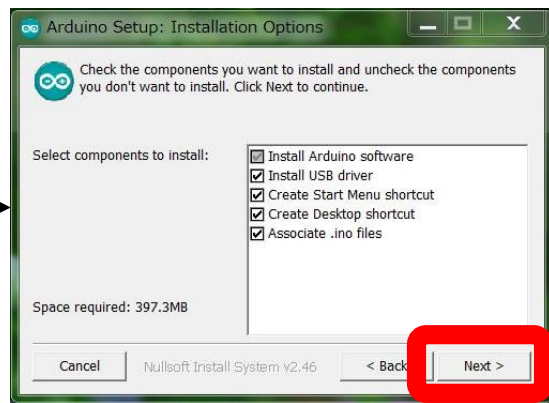
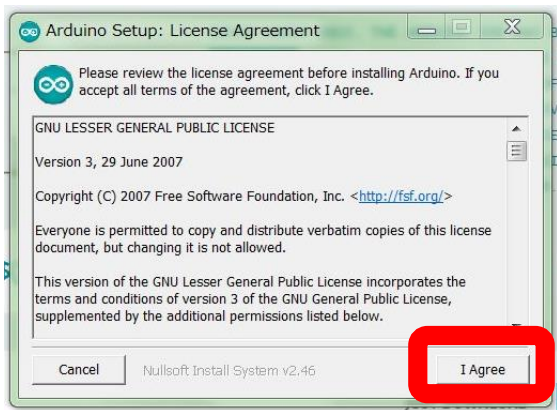
\$50

OTHER

JUST DOWNLOAD

CONTRIBUTE & DOWNLOAD

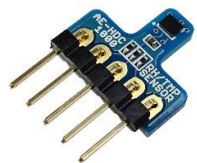
インストールはデフォルト推奨、ドライバーについても全てインストールします。  
#本スライド記載の画像はWindowsの場合になります。



# マイコンおよび プログラムの構築

# 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

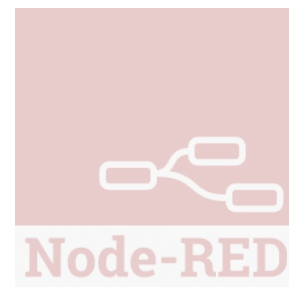


マイコン (Arduino Uno)

sakura.ioの設定



Webサービス連携  
(さくらのクラウド)





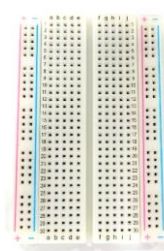
さくらの通信モジュール(アンテナ付)  
+Arduino用シールド&Arduino Uno Rev3



12W級 9V-1.3A  
DCアダプタ



USB2.0  
ケーブル(A-B)

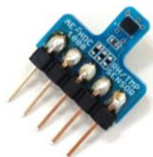


ブレッドボード



ジャンパーコード

本日は使用しません



温湿度センサ  
(HDC1000 or SHT31)



抵抗入りLED  
※必要に応じてご提供



照度センサ  
(GL5537-2)



人感センサ  
(SB412A)



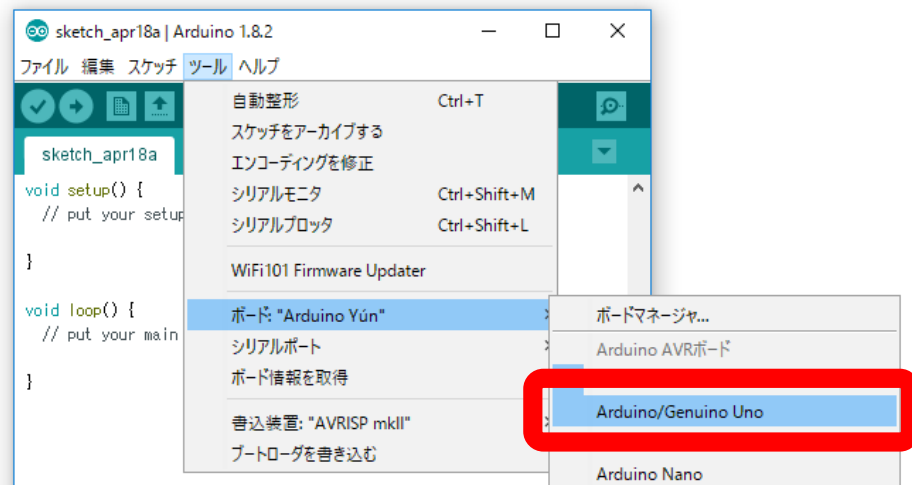
タクトスイッチ  
※必要に応じてご提供



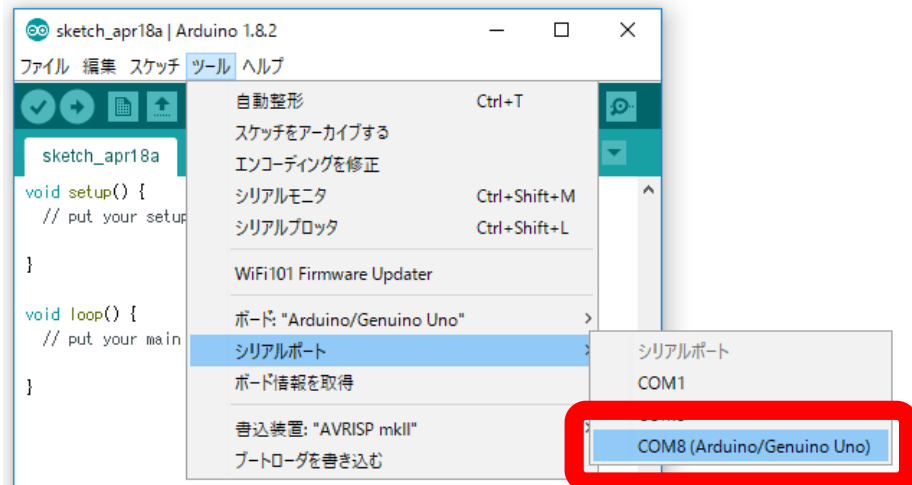
各種抵抗  
※必要に応じてご提供

Arduino IDEが起動したら、Arduino本体をPCに接続します。上部メニューバーから以下2つを設定します。  
ボード: [ツール]→[ボード:"XXX"]から【Arduino/Genuino Uno】を選択します。  
シリアルポート: [ツール]→[シリアルポート]から【〜〜〜 (Arduino/Genuino Uno)】となるものを選択します。

### ボードの選択

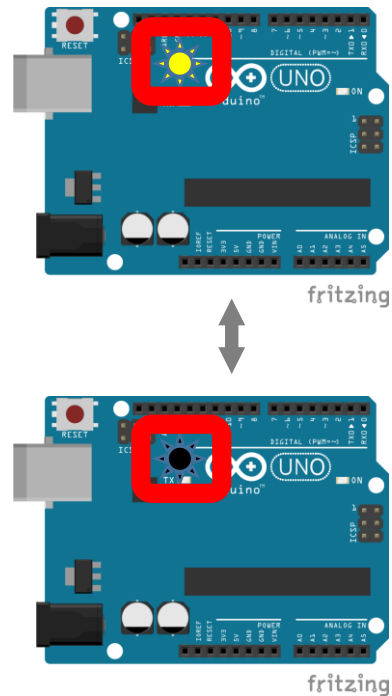
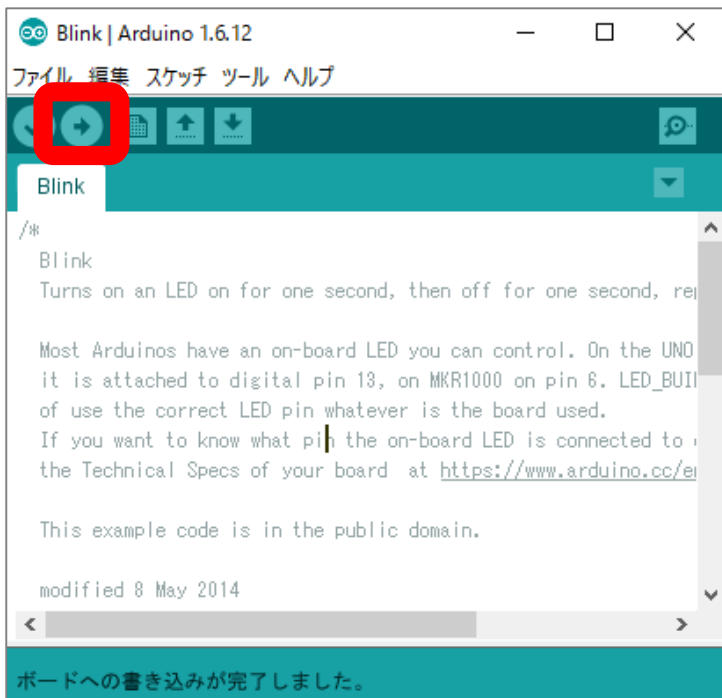


### シリアルポートの選択

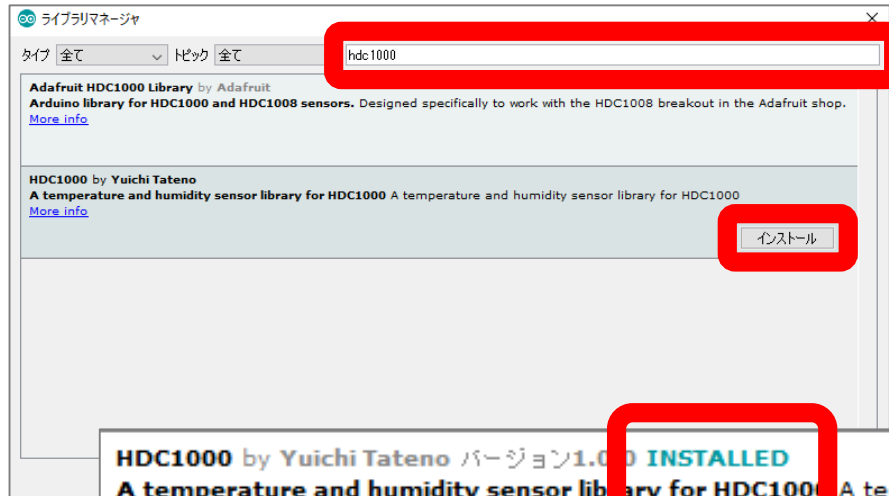




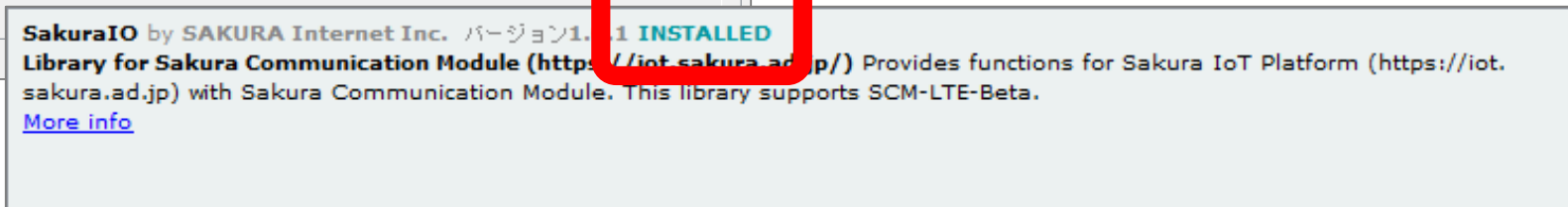
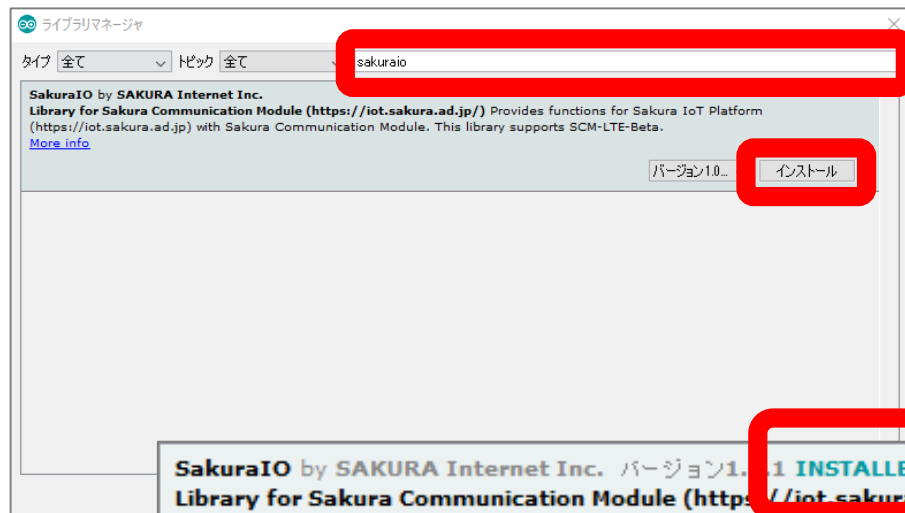
[ファイル]→[スケッチ例]→[01.Basics]→[Blink]を選択し、Blinkスケッチを表示します。  
【→】をクリックしてスケッチをマイコンに書き込み、該当箇所のLEDが点滅状態になることを確認します。  
何らかの問題があった場合、スケッチ下部にオレンジ色のエラーが表示されます。



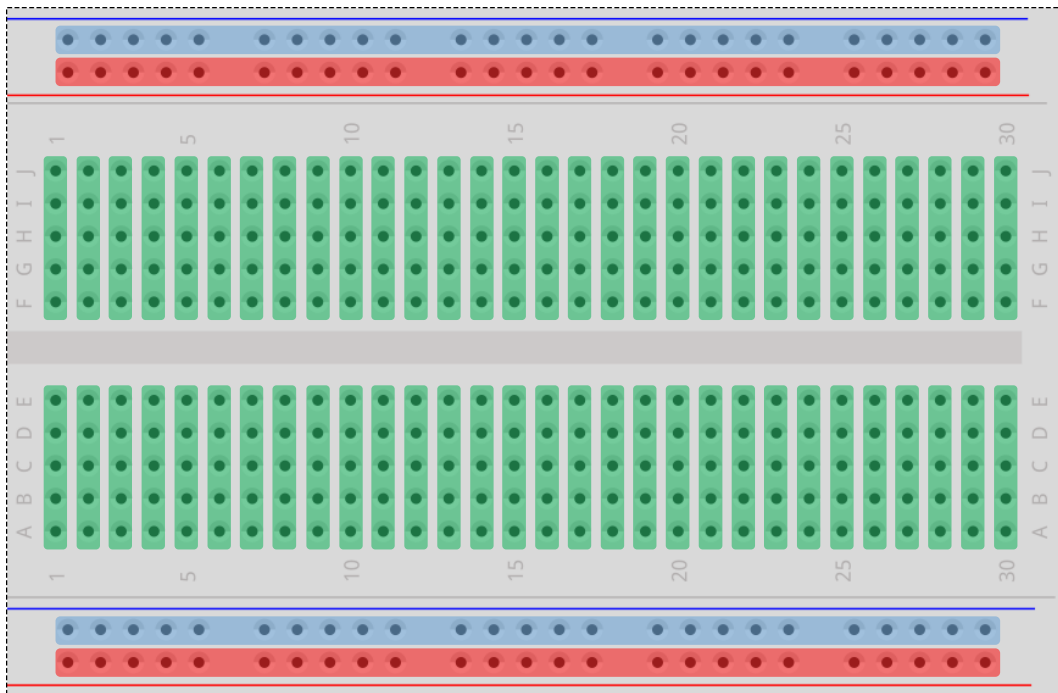
[スケッチ]→[ライブラリをインクルード]→[ライブラリを管理...]をクリックし、  
右上検索窓から【hdc1000】を検索すると、[HDC1000 by Yuichi Tateno]がヒットします。  
インストールをクリックすると該当ライブラリが取り込まれ、[INSTALLED]が表示されます。



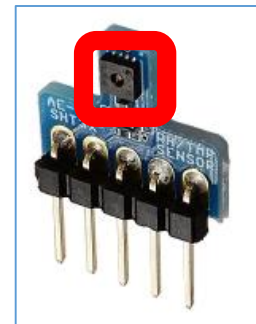
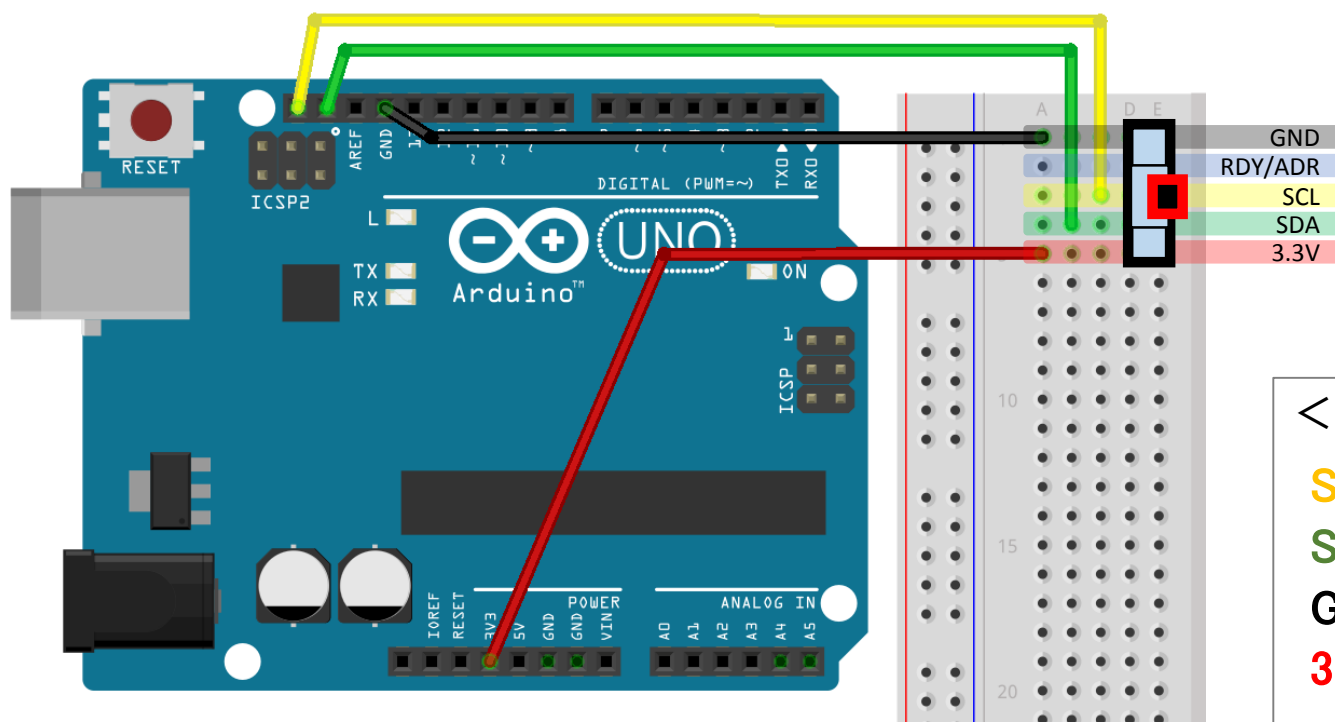
[スケッチ] → [ライブラリをインクルード] → [ライブラリを管理...] をクリックし、  
右上検索窓から【sakuraio】を検索すると、[SakuraIO by SAKURA Internet Inc.] がヒットします。  
最新のVer.を選択のうえインストールをクリックすると該当ライブラリが取り込まれ、[INSTALLED] が表示されます。



ブレッドボードは場所によって接続されている場所が異なります。  
下記の繋がっている部分を意識して配線を行ってください。



図に従い、温湿度センサの向きに注意しながら配線します。**(実際にはArduinoシールドに対して配線します)**  
ブレッドボード側は色で明示された位置であれば、自由に接続しても問題ありません。  
温湿度センサ側のRDY/ADRピンは今回は使用しないため、何も配線しません。

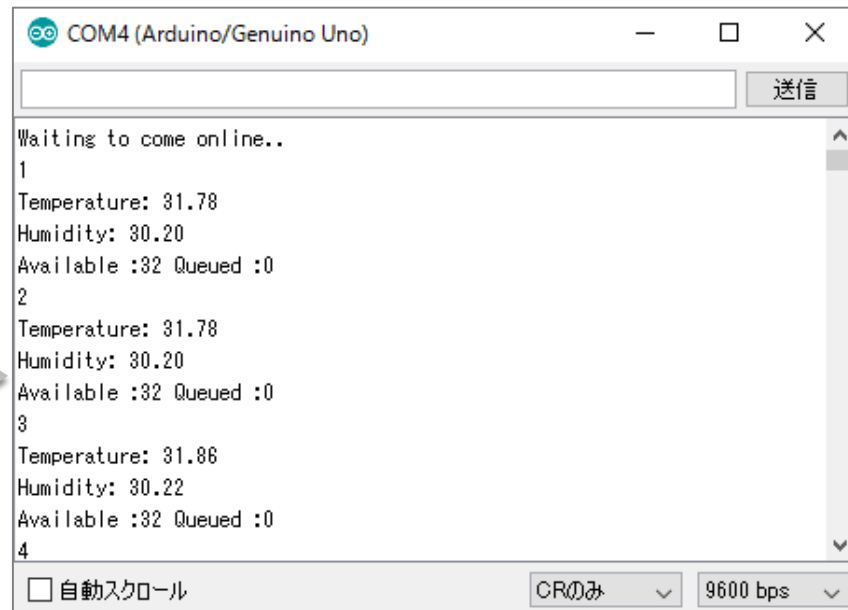
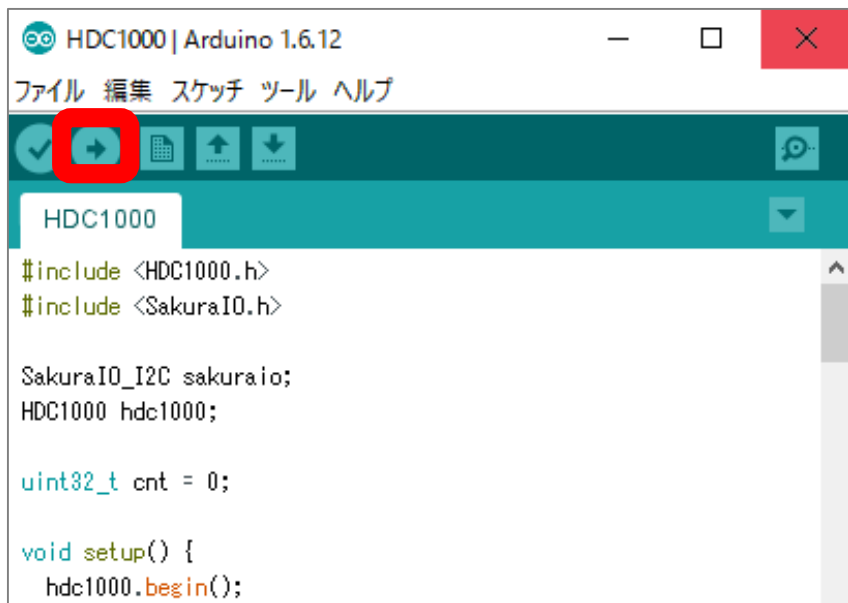


<凡例>

SCL	—
SDA	—
GND	—
3.3V	—



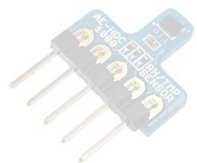
[ファイル]→[スケッチ例]→[SakuraIO]→[**HDC1000**] を選択し、HDC1000スケッチを表示します。  
【→】をクリックし、[ツール]→[シリアルモニタ]より「Waiting to come online」表記の後、  
カウント値、Temperature、Humidityに加え、Available(キューイング可能なチャンネル数)と  
Queued(キューで送信待ちになっているチャンネル数)が表示されることを確認します。



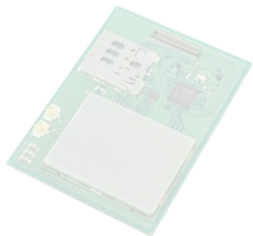
sakura.ioの設定

### 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

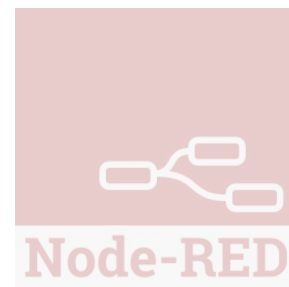


マイコン (Arduino Uno)

### sakura.ioの設定



### Webサービス連携 (さくらのクラウド)



仮想サーバ



sakura.io コントロールパネル (<https://secure.sakura.ad.jp/iot/>) にログインします。  
Googleにて「さくらインターネット iot 開発者」を検索し、開発者向けページの下記ボタンからもアクセスします。

サービスサイト > sakura.io > 開発者向け

コントロールパネル ログイン ➡

既にログイン済みのセッションがない場合、以下画面にて会員認証を求められます。  
会員ID、パスワードを利用してログインします。

## さくらインターネット 会員認証

「会員ID」と「会員メニューのパスワード」をご入力ください  
※ 会員メニューのパスワードはお客さまにてお決めいただいたパスワードです

会員ID

パスワード

**ログイン(認証)**

- 会員メニューのパスワードをお持ちでない方・お忘れの方は[こちら](#)
- 会員IDをお忘れの方は[こちら](#)

sakura.ioでは【プロジェクト】という単位で大枠を構成し、プロジェクト内に複数の【通信モジュール】、【連携サービス】を紐付けていきます。【データストア】や簡易位置情報、ファイル配信といった【オプションサービス】はプロジェクトに対して一つもしくは1セット設定することができます。



初めてコントロールパネルにログインした場合、各種約款への同意を求められます。  
内容をご確認いただいたうえで、[✓同意する]をクリックするとコントロールパネルにアクセスできます。

## 約款および個人情報の取扱いについてのご確認

本サービスを利用するには以下の約款および個人情報の取扱いについてへの同意が必要です。

- [基本約款](#)
- [sakura.io サービス約款](#)
- [sakura.io製品群利用約款](#)
- [sakura.ioポイント約款](#)
- [個人情報の取扱いについて](#)

内容をよくご確認の上、ご同意いただける場合は以下の「同意する」をクリックしてください。

< 同意しない

✓ 同意する

初めはプロジェクトが無い場合、新規にプロジェクトを作成する必要があります。  
まずプロジェクトを作成するため、[+新規プロジェクト]をクリックします。



+ プロジェクト追加

新規プロジェクトの作成画面に遷移します。  
[名称]欄に任意の名前を入力し、[追加]をクリックします。

## 新規プロジェクトの作成

名称

New Project

追加

プロジェクトが作成されました。次に通信モジュールの登録を行います。  
[モジュール登録]のボタンをクリックします。

## New Project #2321

📎 ファイル配信

✎ 編集

🗑 削除

名称

ID

接続

連携サービス

🔗 モジュール登録

+ サービス追加

モジュールの追加画面に遷移します。指定したプロジェクトが選択されていることを確認のうえ、登録用ID、登録用パスワード、および任意の名称を入力して、[追加]ボタンをクリックします。

## モジュールの追加

登録用ID

登録用パスワード

パスワードの表示

名称

プロジェクト

### モジュール追加の際の注意点

モジュールを追加すると、月額60円のモジュール基本料金が課金されます。

追加

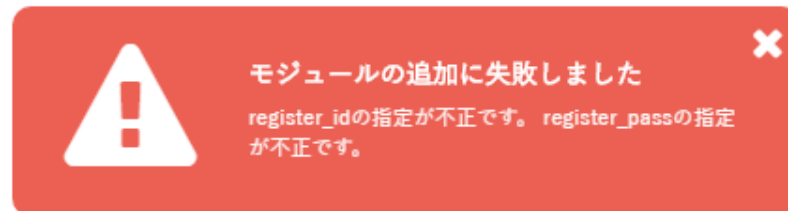


モジュールの追加に成功すると[モジュールを追加しました]というダイアログが表示されます。  
[ホームへ戻る]ボタンをクリックしてホームに戻ります。  
ID/PASSが正しくない、もしくは既に登録されている通信モジュールを追加しようとした場合は  
内容とともに[モジュールの追加に失敗しました]というダイアログが表示されます。

### 登録が成功した場合



### 登録が失敗した場合



登録用ID / 登録用パスワードのいずれかに誤りがあります、再度ご確認ください



別の会員IDに登録されています、過去登録した会員IDにログインし、解除ください

通信モジュールが登録されました。最後に外部への連携サービスを設定します。  
[+サービス追加]のボタンをクリックします。

## New Project #2321

📁 ファイル配信

✎ 編集

🗑 削除

名称

ID

接続

連携サービス

New Module

██████████

オンライン



+ サービス追加

🔗 モジュール登録



追加サービスの選択画面に遷移します。  
今回はWebSocketを作成しますので、[WebSocket]をクリックします。

## 追加サービスの選択

WebSocket

Outgoing Webhook

Incoming Webhook

MQTT Client

DataStore API

AWS IoT

Azure IoT Hub



WebSocketの作成には特に設定事項はありません。  
[名前]欄に任意の名前を入力し、[作成]ボタンをクリックします。

## サービス連携の作成 WebSocket

名前

New Service

作成

連携サービスが登録されました。この後で使用するWebSocketのURLを確認します。  
該当のプロジェクトにて作成された名称の連携サービスをクリックします。

## 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

WebSocketを設定しました、これでコントロールパネルでの準備は完了です。  
ここで表示されるWebSocketのURLはハンズオンの後半で使用しますので書き留めておいてください。

## 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

なお、WebSocketの場合、簡易的に通信モジュールからのデータを確認することができます。  
簡易表示モードでは通信モジュールから受け取ったデータを以下のように確認できます。

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

詳細表示モードに切り替え 切断

時刻	モジュール	チャンネル	型	値
2017-04-19T09:55:32.967182503Z	[REDACTED]	0	f	25.671692
2017-04-19T09:55:32.979182503Z	[REDACTED]	1	f	28.112793
2017-04-19T09:55:32.991182503Z	[REDACTED]	2	l	4
2017-04-19T09:55:27.895144714Z	[REDACTED]	0	f	25.722046
2017-04-19T09:55:27.907144714Z	[REDACTED]	1	f	28.692627
2017-04-19T09:55:27.919144714Z	[REDACTED]	2	l	3

→温度

→湿度

→カウント値

[詳細表示モードに切り替え] をクリックすると詳細表示モードに遷移します。  
時刻やモジュールは同一ですが、データの内容によりタイプやペイロードが異なります。

複数チャンネルある場合、  
それぞれの最終データを表示

クリックでログ出力を  
開始/停止

タイプはいくつかの種類に分かれる

- channels : 通信モジュールからのデータ
- connection : 接続の開始/終了
- location : 簡易位置情報
- keepalive : WebSocketセッション確認

ペイロードには受け取った  
データのpayload部が表示

詳細表示モードに切り替え

**チャンネル毎の最終到着データ**

Ch: 0	Ch: 1	Ch: 2
25.722046	26.171875	2

ログ 50件まで

時刻	モジュール	タイプ	ペイロード
2017-04-19T10:23:54.897157339Z		channels	{ "channel": 0, "type": "f", "value": 25.722046, "datetime": "2017-04-19T10:23:54.862158727Z" } { "channel": 1, "type": "f", "value": 26.171875, "datetime": "2017-04-19T10:23:54.874158727Z" } { "channel": 2, "type": "l", "value": 2, "datetime": "2017-04-19T10:23:54.886158727Z" }
2017-04-19T10:23:53.385936233Z		channels	{ "channel": 0, "type": "f", "value": 25.76233, "datetime": "2017-04-19T10:23:53.350937578Z" } { "channel": 1, "type": "f", "value": 26.171875, "datetime": "2017-04-19T10:23:53.362937578Z" } { "channel": 2, "type": "l", "value": 1, "datetime": "2017-04-19T10:23:53.374937578Z" }
2017-04-19T10:23:53.1004698Z		connection	{ "is_online": true }
2017-04-19T10:23:52.735071228Z		keepalive	



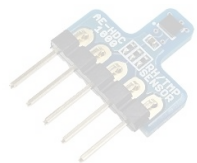
Webへのデータ連携  
(さくらのクラウド)

マイコンおよび  
プログラムの構築

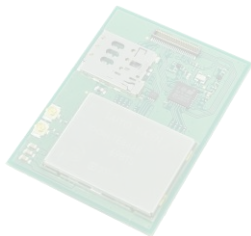
sakura.ioの設定

Webサービス連携  
(さくらのクラウド)

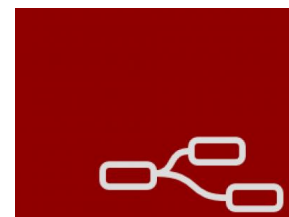
温湿度センサ  
(SHT31/HDC1000)



さくらの通信  
モジュール



マイコン (Arduino Uno)



Node-RED



仮想サーバ

さくらのクラウドコントロールパネル (<https://secure.sakura.ad.jp/cloud/>) にログインします。  
「さくらインターネット会員としてログイン:」に会員ID、パスワードを入力してログインすることができます。  
ハンズオンでは既にさくらのクラウドのユーザは作成済みとなります。

The screenshot shows the login page for Sakura Cloud. It is divided into two main sections. The left section, titled 'さくらインターネット会員としてログイン:' (Login as a Sakura Internet Member), features a red-bordered box around the '会員ID' (Member ID) and 'パスワード' (Password) input fields, and a blue 'ログイン' (Login) button. Below these fields are links for '新規登録はこちら' (New registration here), '会員IDを忘れた' (Forgot member ID), and 'パスワードを忘れた' (Forgot password), and a checkbox for '会員IDを保存' (Save member ID). The right section, titled 'さくらのクラウドユーザとしてログイン:' (Login as a Sakura Cloud User), has input fields for 'ユーザコード' (User Code) and '会員ID' (Member ID) with an '@' symbol between them, and a 'パスワード' (Password) field with a green 'ログイン' (Login) button. It also includes a checkbox for 'ユーザコードと会員IDを保存' (Save user code and member ID) and a note: 'ユーザコード・パスワードが分からない場合はさくらインターネット会員としてログインするか、ユーザの管理者にお問い合わせください。' (If you don't know your user code or password, log in as a Sakura Internet member or contact the user administrator). The page footer contains '個人情報保護ポリシー 約款' (Privacy Policy Terms) on the left and '© 2017 SAKURA Internet Inc.' on the right.

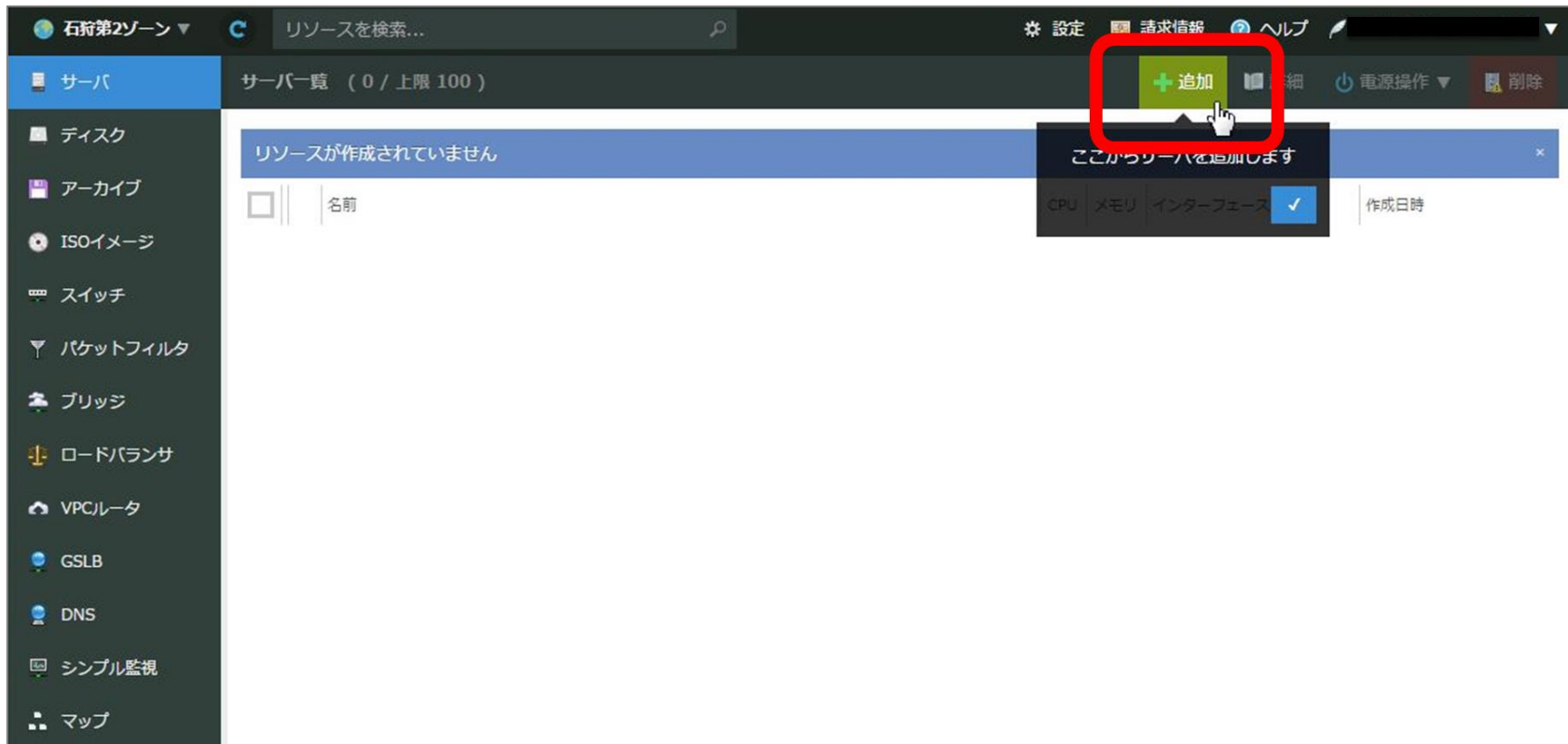
さくらインターネット会員としてログインするとアカウントの選択を求められます。  
自身が利用するアカウントを選択します。  
アカウントを作成していない場合は上記タブからアカウントを選択し作成する必要があります。



ユーザでログインができれば[さくらのクラウド(IaaS)]をクリックします。

The screenshot shows the Sakura Cloud control panel home page. At the top, there is a green navigation bar with links for 'さくらのクラウド ホーム', 'サービス', 'アカウント', 'ユーザ', '2段階認証', and 'イベントログ'. On the right side of the bar, there is a language selector set to '日本語' and a 'ログアウト' button. Below the navigation bar, the current account is identified as 'Seminar2016 (Seminar2016)'. A light blue banner prompts the user to '管理するサービスを選択してください'. Below this banner, there are five service selection tiles: 'さくらのクラウド (IaaS)' (highlighted with a red circle), 'オブジェクトストレージ', 'グローバル', 'カタログ', and 'シンプル監視'. At the bottom left, there is a '請求情報' (Billing Information) tile with a yen symbol icon.

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ことがあります。



デフォルトではサーバの作成は細かい設定が不要な「シンプルモード」で作成できます。  
スタートアップスクリプトを利用する場合は右上の【シンプルモード】のチェックを外します。

The screenshot shows the 'Server Addition' (サーバ追加) page in the Sakura Internet control panel. The 'Simple Mode' (シンプルモード) checkbox is checked and highlighted with a red box. Below this, users can select a disk image (ディスクイメージ) and a server plan (サーバプラン).

**ディスクイメージを選択 \***

Unix / Linux	Windows	パッケージ	マイアーカイブ	マイディスク	
 CentOS 6.8 64bit	 Ubuntu Server	 Debian GNU/Linux	 FreeBSD	 CoreOS	 VyOS

管理ユーザ名は「root」です。  
サーバ作成後、rootユーザでログインしてください。

**サーバプランを選択**

¥1,522/月 ¥76/日 ¥7/時 1 GB / 1 仮想コア	¥3,240/月 ¥162/日 ¥16/時 2 GB / 2 仮想コア	¥4,860/月 ¥243/日 ¥23/時 4 GB / 2 仮想コア	¥8,100/月 ¥405/日 ¥39/時 4 GB / 4 仮想コア	¥11,340/月 ¥567/日 ¥56/時 8 GB / 4 仮想コア
---	---	---	---	--

全てのアイテムから選択...



サーバプランでは仮想サーバに割り当てるCPUとメモリの量を指定します。  
仮想コアは【1】を、メモリは【1GB】を、それぞれ選択します。

1. サーバプラン

仮想コア

1  2  3  4  5  6  8  10  12

メモリ

1GB  2GB  3GB  4GB  5GB

[サーバプラン一覧から選択](#)

- プラン/1Core-1GBが選択されました





2. ディスクでは使用するディスクの種類やサイズ、インストールイメージを選択します。  
アーカイブ選択のみ【CentOS 7.x 64bit #xxxxxxxxxxxx】を選択し、後はデフォルトとします。

新規ディスクを作成  既存ディスクを接続  ディスクレス (なし)

ディスクプラン

SSDプラン  標準プラン

ディスクソース

アーカイブ  マイアーカイブ  マイディスクをコピー  ブランク (空のディスク)

さくらにて用意した初期設定済みOSイメージはアーカイブとして提供されています

アーカイブ選択

CentOS 7.3 (1611) 64bit #112900084256

管理ユーザ名は「root」です。  
サーバ作成後、rootユーザでログインしてください。

全サイズ対応：選択されたディスクプランに合わせてパーティションサイズが最適化されます。

ディスクサイズ

20GB

別のストレージに収容する  
指定されたディスクとは別のストレージにディスクを作成します

準仮想化モードを使う (Virtio)  
有効にすると、ディスクアクセスが高速になります。別途ドライバが必要になる場合があります。

2. ディスク



3.NICではネットワークに関する設定を指定します。  
今回はすべてデフォルトの値を使用しますので変更は不要です。

## 3. NIC

インターネットに接続  スイッチに接続  切断

NICは、後からも追加・変更することができます

インターネットに接続

1IP 共有セグメント / 100Mbps ベストエフォート ▼

準仮想化 モードを使う (Virtio)

有効にすると、@virtio-net-pciタグ が設定され、高速に通信できるようになります。別途ドライバが必要になる場合があります。

▼ パケットフィルタ選択

- ▼



4. ディスクの修正ではOSに関する設定値を指定します。  
管理ユーザのパスワードおよびホスト名は任意の値を入力します。  
公開鍵は今回のハンズオンでは使用しないためデフォルトの【なし】を使用します。

ディスクの修正をする

ホスト名、パスワード、公開鍵の他、インターフェース設定に関するディスク内容が修正されます。※一部非対応の構成・OSがあります

管理ユーザのパスワード

- パスワード強度：強い

ホスト名

サーバを一括作成した場合は、ホスト名、リソース名の後ろに連番が付与されます（例：hostname-01, hostname-02...）

公開鍵

なし  入力  選択

4. ディスクの修正



[スタートアップスクリプト]で【shell】を選択のうえ、[配置するスタートアップスクリプト]で【[public] Node-RED #xxxxxxx】を選択します。オプションのWebUIポート番号には【80】を入力します。

## 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。

## スタートアップスクリプト オプション

### Node-REDのWeb UIポート番号

80

- ディスクの内容は修正されます



5. シンプル監視はさくらのクラウドで提供する死活監視のサービスとなります。  
本件では使用しないため、デフォルトのチェックなしで進めます。

### 5. シンプル監視

シンプル監視を有効にする

作成するサーバにシンプル監視を適用します。無料をご利用いただけます。



6.サーバの情報はコントロールパネル上で管理するための情報を記述する項目となります。本件では名前に判別がつくような任意の値を入力します。

6. サーバの情報

名前

test

サーバを一括作成した場合は、ホスト名、リソース名の後ろに連番が付与されます (例: hostname-01, hostname-02...)

説明

1~512文字

タグ

...

アイコン

- ▼



7. その他のオプションおよび作成数はすべてデフォルトの値を使用します。  
内容を確認し、問題がなければ【作成】をクリックします。

7. その他のオプション

仮想サーバ上のキーボードをUS配列として設定する\* (@keyboard-usタグ)

\*: 仮想サーバ上の設定に合わせてください

作成後すぐに起動

作成数 \* 1

作成



操作確認のダイアログにて、作成しても良いか改めて確認されますので、問題がなければ【作成】をクリックすると指定したサーバやディスクの作成を開始します。ステータスが全て成功になればサーバの作成が完了となります。

### 操作確認

課金対象のリソースを一つまたは複数作成します（料金は作成時より個別に計算されます）

内訳を表示    キャンセル    **作成**

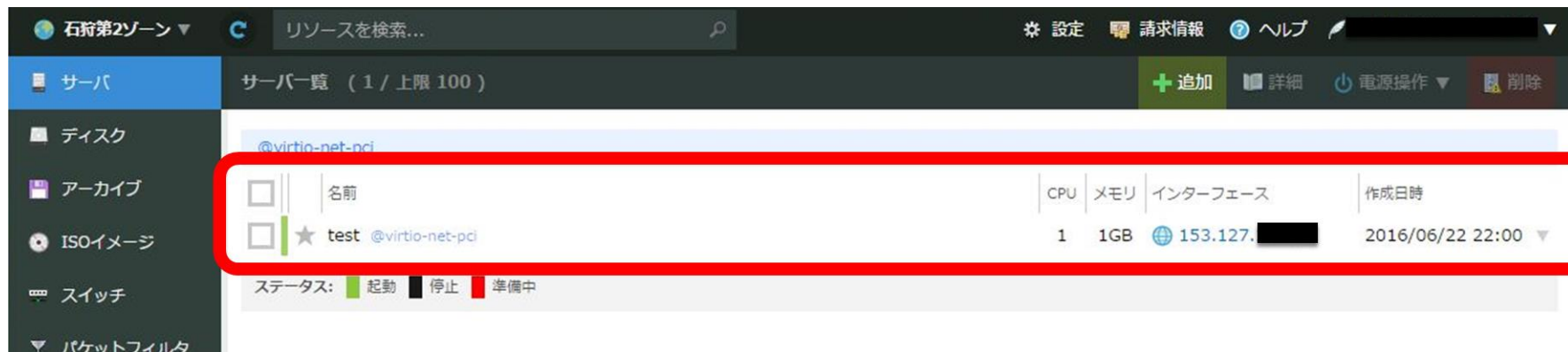
### サーバ追加 - test

名前	メソッド	リソース	ステータス
サーバ: 作成	POST	cloud/1.1/server	<b>要求</b>
ディスク: 作成	POST	cloud/1.1/disk	待機中
ディスク: 準備完了を待機	GET	cloud/1.1/disk/:diskId	待機中
ディスクの修正	PUT	cloud/1.1/disk/:diskId/config	待機中
すぐに起動	PUT	cloud/1.1/server/:serverId/power	待機中

中断    閉じる



サーバが作成されました。【 [http://<IPアドレス>:<指定したWeb UIポート番号>/](http://<IPアドレス>:<指定したWeb UIポート番号>) 】にWebブラウザよりアクセスします。WebUIポートに80番ポートを指定した場合はポート番号を省略可能です。



石狩第2ゾーン リソースを検索...

設定 請求情報 ヘルプ

サーバ サーバー一覧 (1 / 上限 100) + 追加 詳細 電源操作 削除

ディスク

アーカイブ

ISOイメージ

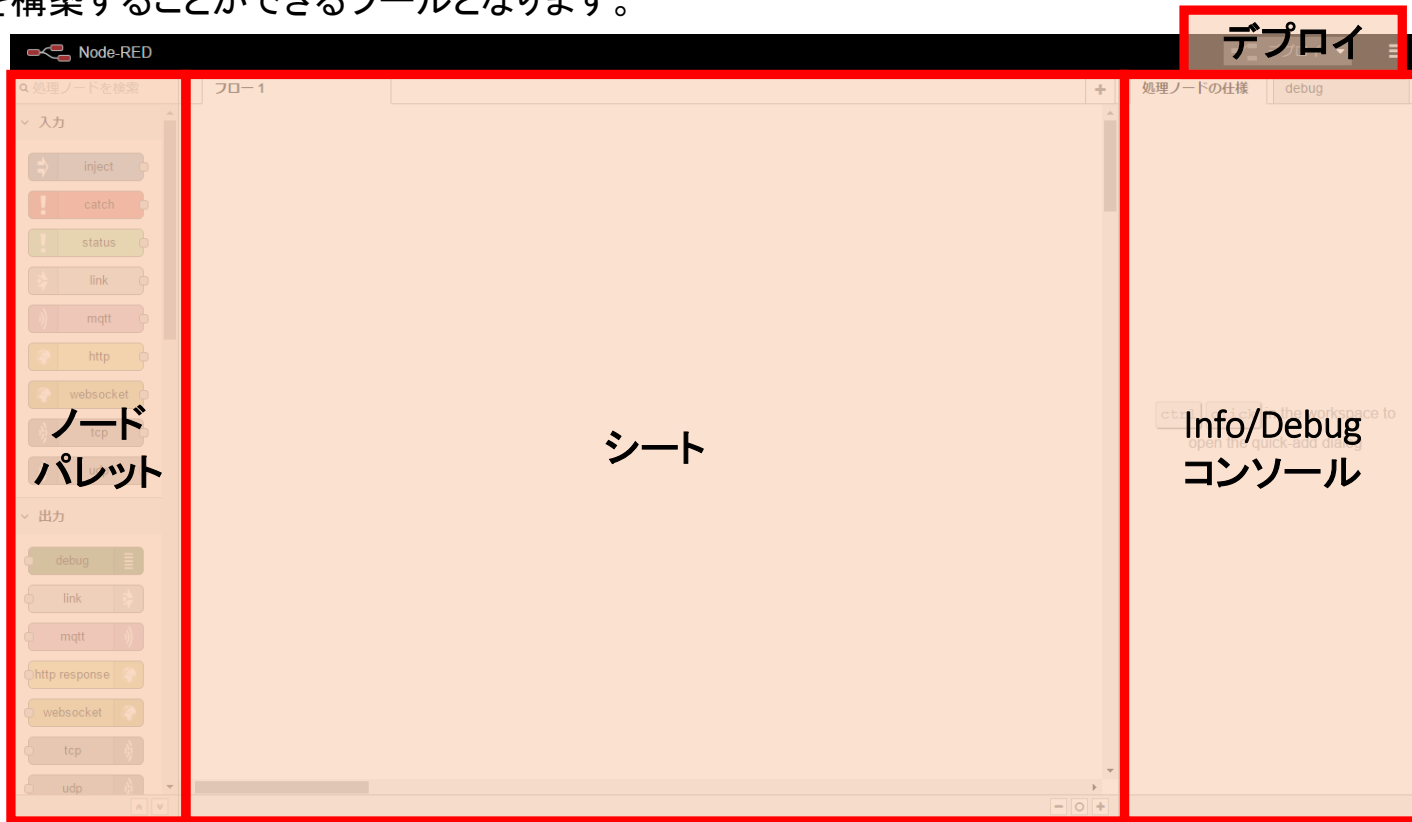
スイッチ

パケットフィルタ

名前	CPU	メモリ	インターフェース	作成日時
test @virtio-net-pci	1	1GB	153.127. [REDACTED]	2016/06/22 22:00

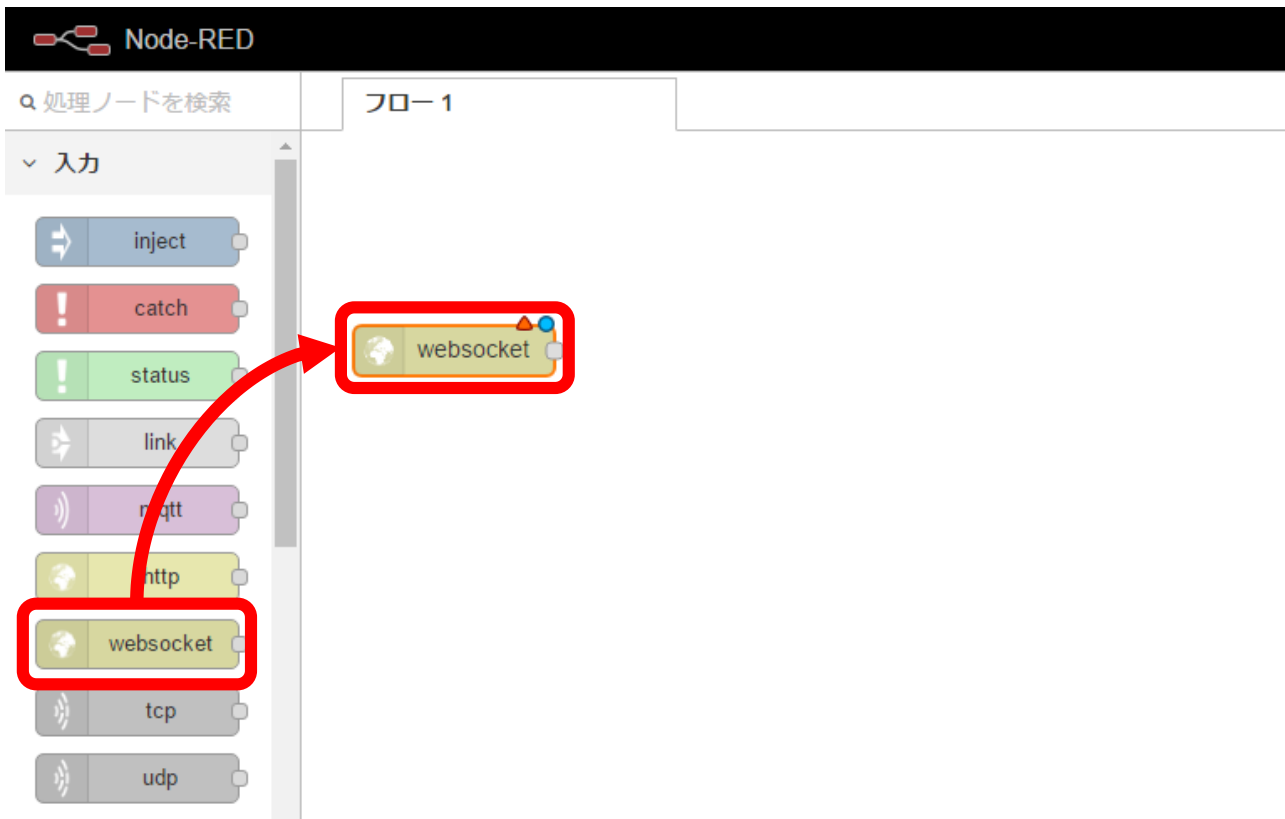
ステータス: 起動 停止 準備中

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。





まずはWebSocketからのデータを受け取るノードを追加します。  
ノードパレットの入力から「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。  
Typeは「接続」、Nameは「任意の名称」を入力の上、URLの行にある鉛筆マークをクリックします。

フロー 1

websocket

websocket in 処理ノードを編集

削除 中止 完了

Type Connect to

URL 新規に websocket-client を追加。。。

Name sakuraio-websocket



指定するURL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URL部分にペーストします。

## 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



URL部分はコンパネからのペーストを行い、ドロップダウンの項目については「ペイロードを送信/受信」を選択し、【追加】をクリックします。

フロー 1

websocket

websocket in > 新規に websocket-client 処理ノードの設定を追加

中止 追加

URL

wss://api.sakura.io/ws/v1/ [REDACTED]

Send/Receive payload

URL should use ws:// or wss:// scheme and point to an existing websocket listener.

By default, `payload` will contain the data to be sent over, or received from a websocket. The client can be configured to send or receive the entire message object as a JSON formatted string.



【完了】をクリックするとwebsocketノードへの設定が反映され、設定不備がない場合はノードの三角マークが消えます。WebSocketノードへの設定が反映されましたが、この時点ではsakura.ioからデータは入ってきていません。続いてはデータを表示するためのDebugノードを作成します。

websocket in 処理ノードを編集

削除 中止 **完了**

Type

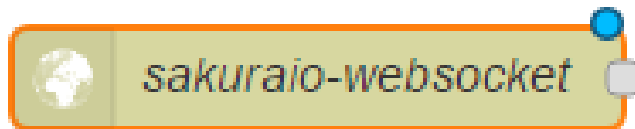
Connect to

URL

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

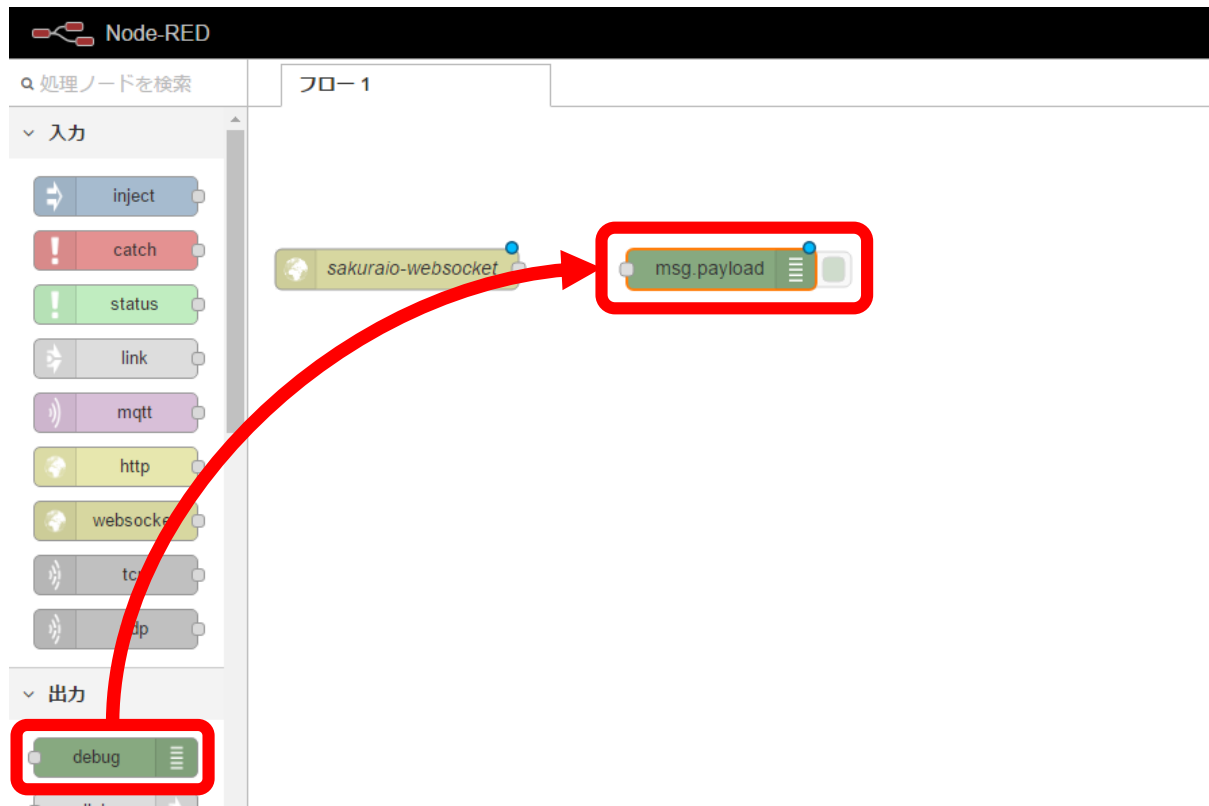
Name

sakuraio-websocket





次に、ノードパレットの出力から「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」に名前が変わります。特に設定は不要です。

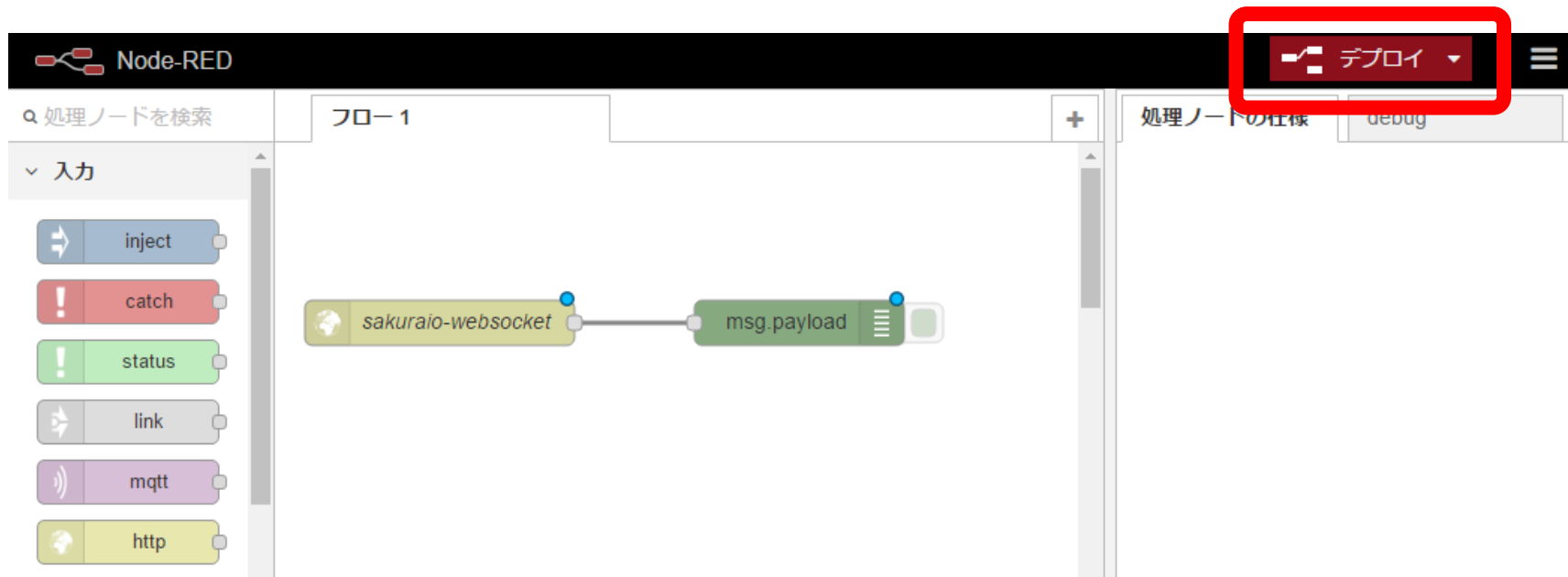




各ノードの動作を繋げるために、WebSocketノード右端とDebugノード左端をドラッグ&ドロップで繋ぎます。



各ノードを接続し、準備が完了したら、右上部の【デプロイ】をクリックします。  
デプロイが完了するとデプロイ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、コンソールのデバッグ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。デバッグ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。

The screenshot shows the Node-RED web interface. On the left, the 'Inputs' panel contains various nodes, with 'sakuraio-websocket' highlighted and showing a 'connected' status. This node is connected to a 'msg.payload' node. A red box highlights the 'sakuraio-websocket' node and its connection. Another red box highlights the green square icon on the 'msg.payload' node. On the right, the 'debug' console is open, displaying a stream of JSON messages. A large red box encompasses the entire debug console area. The messages are as follows:

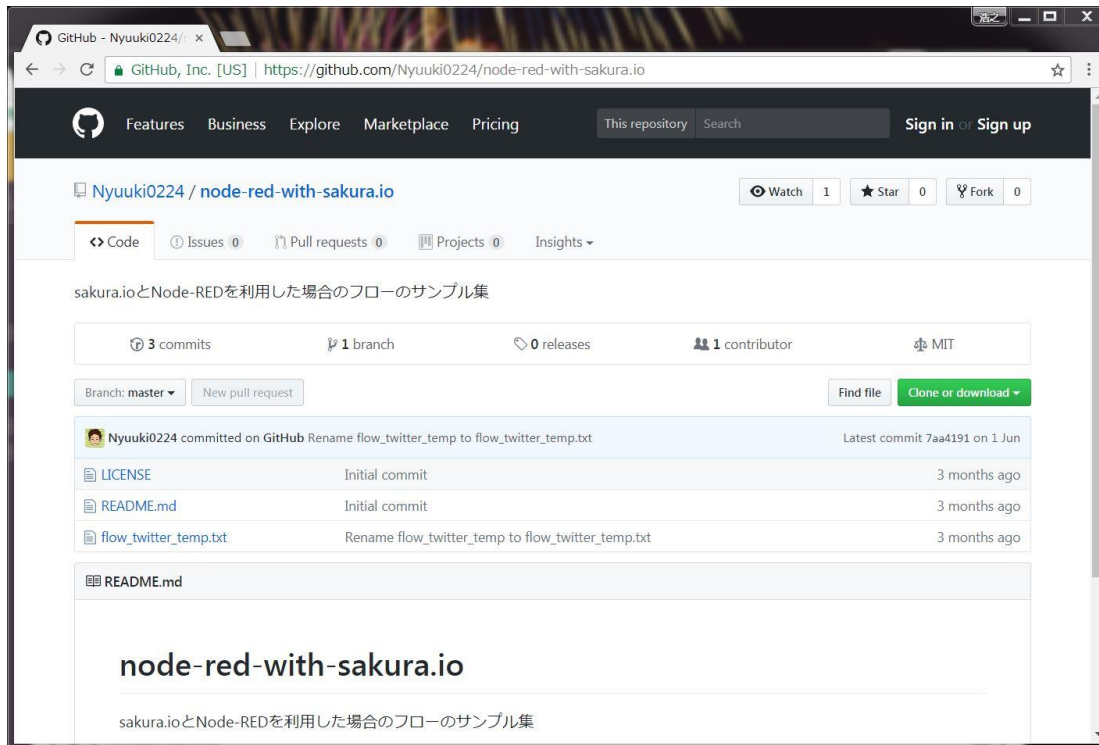
```
2017/5/31 12:51:51 node: 3d6d2a19.274956
msg.payload: string[367]
{"module": "██████████", "type": "channels", "datetime": "2017-05-31T03:51:53.293714244Z", "payload":
{"channels": [{"channel": 0, "type": "f", "value": 26.76941, "datetime": "2017-05-31T03:51:53.258726713Z"},
{"channel": 1, "type": "f", "value": 36.77368, "datetime": "2017-05-31T03:51:53.270726713Z"},
{"channel": 2, "type": "I", "value": 81344, "datetime": "2017-05-31T03:51:53.282726713Z"}]}}
```

```
2017/5/31 12:51:52 node: 3d6d2a19.274956
msg.payload: string[367]
{"module": "██████████", "type": "channels", "datetime": "2017-05-31T03:51:54.303640749Z", "payload":
{"channels": [{"channel": 0, "type": "f", "value": 26.76941, "datetime": "2017-05-31T03:51:54.268641689Z"},
{"channel": 1, "type": "f", "value": 36.77368, "datetime": "2017-05-31T03:51:54.280641689Z"},
{"channel": 2, "type": "I", "value": 81345, "datetime": "2017-05-31T03:51:54.292641689Z"}]}}
```

```
2017/5/31 12:51:53 node: 3d6d2a19.274956
msg.payload: string[368]
{"module": "██████████", "type": "channels", "datetime": "2017-05-31T03:51:55.312609326Z", "payload":
{"channels": [{"channel": 0, "type": "f", "value": 26.759338, "datetime": "2017-05-31T03:51:55.277610286Z"},
{"channel": 1, "type": "f", "value": 36.77368, "datetime": "2017-05-31T03:51:55.289610286Z"},
{"channel": 2, "type": "I", "value": 81346, "datetime": "2017-05-31T03:51:55.301610286Z"}]}}
```

取得した温度情報を元にメッセージを生成し、Twitterアカウントへ投稿するフローを入手します。  
「github nyuuki0224」で検索し、検索結果からGitHubのサイトへ移動します。

<https://github.com/Nyuuki0224/node-red-with-sakura.io>



<https://github.com/Nyuuki0224/node-red-with-sakura.io>

ファイルリストから「flow\_twitter\_temp.txt」をクリックし、ソースコード表示画面右上の「Raw」をクリックするとフローサンプルがプレーンテキストで表示されますので、全部コピーします。

Nyuuki0224 committed on GitHub Rename flow\_twitter\_temp to flow\_twitter\_temp.txt

LICENSE	Initial commit
README.md	Initial commit
<b>flow_twitter_temp.txt</b>	Rename flow_twitter_temp to flow_twitter_temp.txt

Nyuuki0224 Rename flow\_twitter\_temp to flow\_twitter\_temp.txt 7aa4191 on 1 Jun

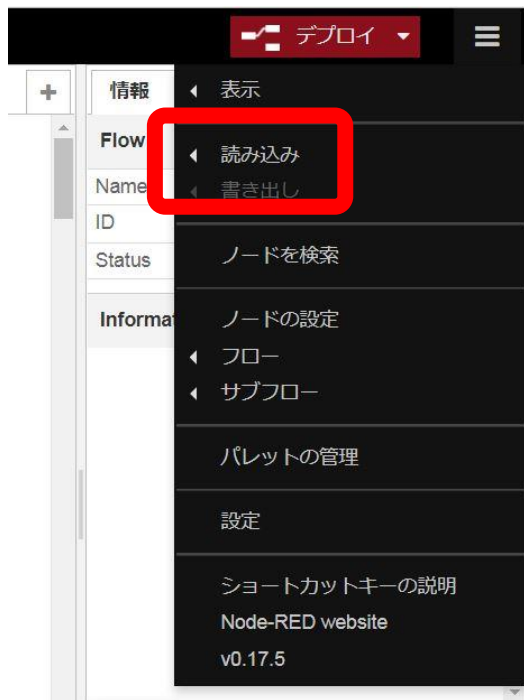
1 contributor

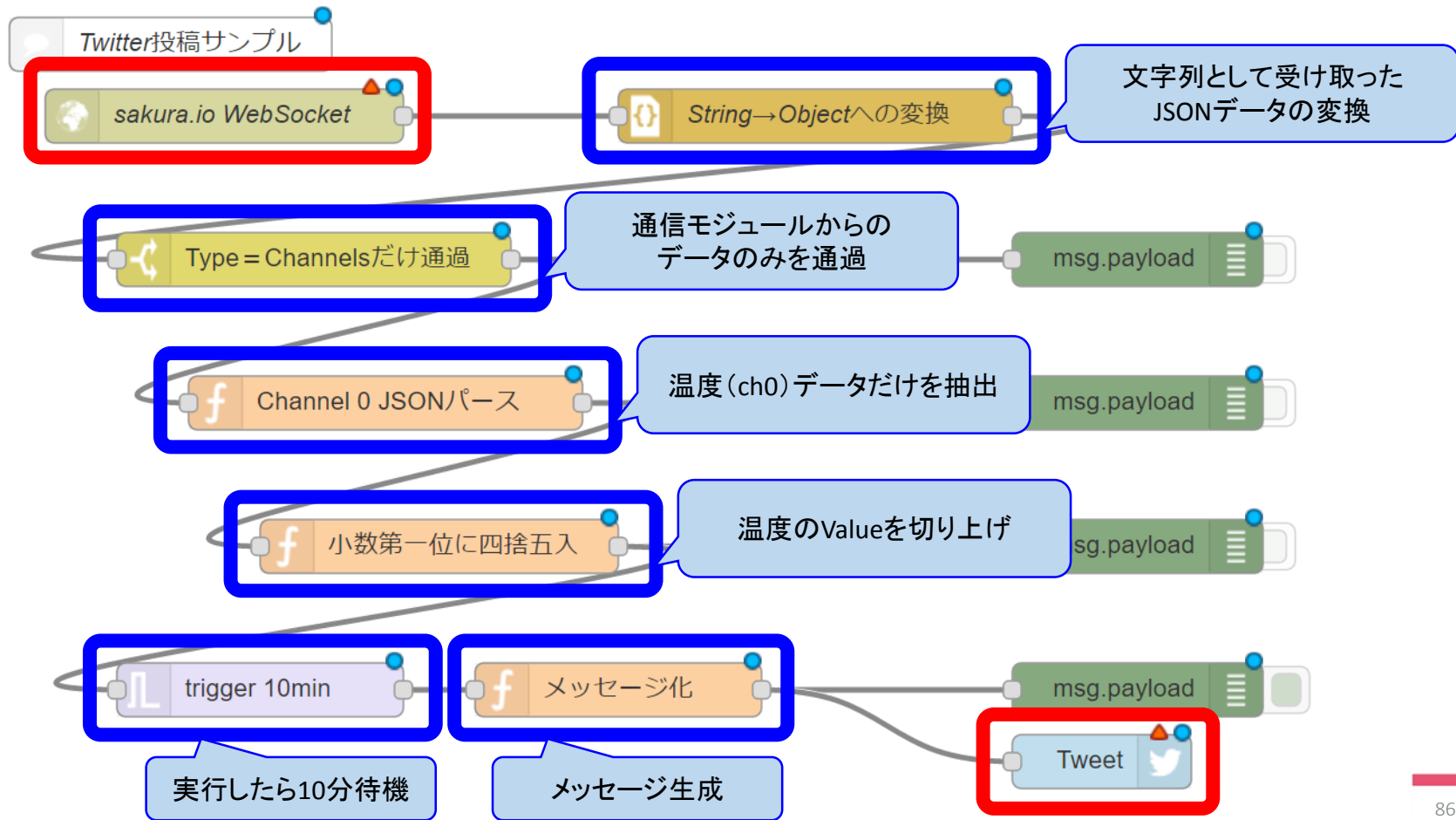
201 lines (200 sloc) | 4.88 KB

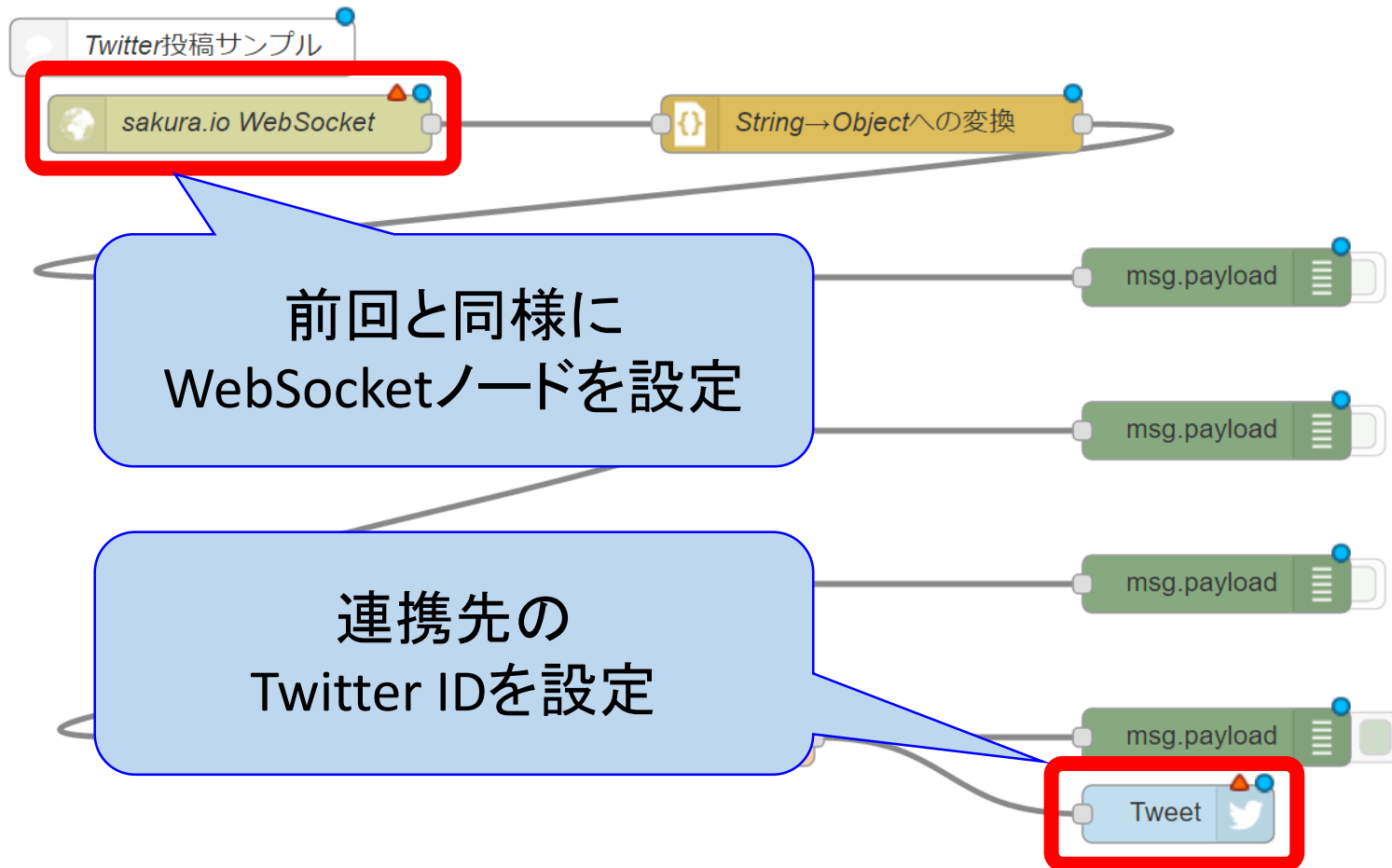
Raw Blame History

```
1 [
2   {
3     "id": "7f4dd36d.81291c",
4     "type": "tab",
5     "label": "Twitter投稿"
6   },
```

Node-RED右上の【☰】→【読み込み】→【クリップボード】を選択し、画面内のポップアップに、先ほどコピーしたフローサンプルをペーストして、「読み込み」をクリックして保存します。  
読み込まれたフローは「Twitter投稿」というタブに表示されます。









フローサンプル右下のTwitterノードをダブルクリックし、設定画面に移ります。  
Twitter IDは「新規にtwitter-credentialsを追加」に設定し、その右にある鉛筆マークをクリックします。



「Twitterの認証を行うため、ここをクリックしてください」をクリックすると、Twitterの連携アプリ認証画面に遷移します。



TwitterのIDとパスワードを入力し、「連携アプリを認証」をクリックします。  
「Authorised - you can close this window and return to Node-RED」が表示されたら認証に成功しています。



アカウント作成

### Node RED にアカウントの利用を許可しますか？

ユーザー名、またはメールアドレス

パスワード

保存する - パスワードを忘れた場合はこちら

**連携アプリを認証** キャンセル



**Node-RED**

Node RED

nodered.org

Node-RED Twitter node

Node-REDの画面に戻って「追加」ボタンを押し、「完了」ボタンを押すと連携の設定が完了します。

The image displays two panels from the Node-RED interface. The top panel, titled "twitter out > 新規に twitter-credentials ノードの設定を追加", shows a configuration form with a "Twitter ID" field containing a redacted value. The "追加" (Add) button is highlighted with a red box. The bottom panel, titled "twitter out ノードを編集", shows the same configuration form with an additional "名前" (Name) field set to "Tweet". The "完了" (Complete) button is highlighted with a red box.

各ノードを接続し、準備が完了したら、右上部の【デプロイ】をクリックします。  
デプロイが完了すると処理が開始され、温度情報付きのメッセージがTwitterに投稿されます。(10分毎)

The screenshot shows the Node-RED interface with a flow named "Twitter投稿". The flow consists of the following nodes:

- SAKURA IoT Platform WebSocket (Input)
- String→Objectへの変換 (Function)
- Type = Channelsだけ通過 (Function)
- Channel 0 JSONパース (Function)
- 小数第一位に四捨五入 (Function)
- trigger 10min (Trigger)
- メッセージ化 (Function)
- msg.payload (Output)
- Tweet (Output)

The "Deploy" button in the top right corner is highlighted with a red box. The right sidebar shows the flow's execution history with two messages:

```
2017/8/25 17:21:43 node: d2eab392b7adb8
msg.payload: string[368]
"
{"module":"uPUsHfYk2pgl","type":"channels","date":
08-25T08:21:43.624796419Z,"payload":
{"channels":
[{"channel":0,"type":"f","value":26.235857,"date":
08-25T08:21:43.590797957Z},
{"channel":1,"type":"f","value":60.302734,"date":
08-25T08:21:43.602797957Z},
{"channel":2,"type":"1","value":2191,"date":
08-25T08:21:43.613797957Z}]}]}

2017/8/25 17:21:49 node: d2eab392b7adb8
msg.payload: string[367]
"
{"module":"uPUsHfYk2pgl","type":"channels","date":
08-25T08:21:48.683657501Z,"payload":
{"channels":
[{"channel":0,"type":"f","value":26.26587,"date":
08-25T08:21:48.659658689Z},
{"channel":1,"type":"f","value":60.302734,"date":
08-25T08:21:48.671658689Z},
{"channel":2,"type":"1","value":2192,"date":
08-25T08:21:48.683658689Z}]}]}


```



連携に成功するとこのようなメッセージがツイートされます。

## ツイート ツイートと返信



法林浩之(バースト用) @hourin\_burst · 5分

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26.5度だよ。やっと正式サービスが始まりました、今後ともよろしくお願ひします！ #sakuraio #さくらインターネット



法林浩之(バースト用) @hourin\_burst · 45分

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26.2度だよ。やっと正式サービスが始まりました、今後ともよろしくお願ひします！ #sakuraio #さくらインターネット



そこに、さくら

質疑応答



最後に

ハンズオンに使用したさくらの通信モジュールは必要に応じて登録を解除します。  
接続ステータス横の歯車マークをクリックします。

New Project #2321

ファイル配信 編集 削除

名称	ID	接続	
New Module	[REDACTED]	オフライン	

連携サービス

New Service websocket

+ サービス追加

モジュール登録

モジュールの設定画面に遷移します。下部の[登録解除]ボタンをクリックします。

## モジュールの設定

Name

New Module

ID

[Redacted]

Model

SCM-LTE-BETA

Serial

[Redacted]

プロジェクト

2321 - New Project

登録解除

変更

通信モジュールの登録を解除してよいか確認されます。  
問題ない場合は再度[登録解除]ボタンをクリックします。

## モジュールの登録解除

モジュールの登録を解除します。

登録解除したモジュールは約款で定める期間が経過するとプラットフォームに再登録できなくなります。 <https://sakura.io/agreement/>

よろしければ[登録解除]をクリックください。

登録解除

モジュールの登録が解除され、表示から削除されました。  
次は連携サービスとともにプロジェクトを削除します。  
プロジェクト右上の[削除]マークをクリックします。

New Project #2321

ファイル配信 編集 **削除**

名称	ID	接続
		<a href="#">モジュール登録</a>

連携サービス

New Service **websocket**

[+ サービス追加](#)

プロジェクトを削除してよいか確認されます。  
問題ない場合は再度[削除]ボタンをクリックします。

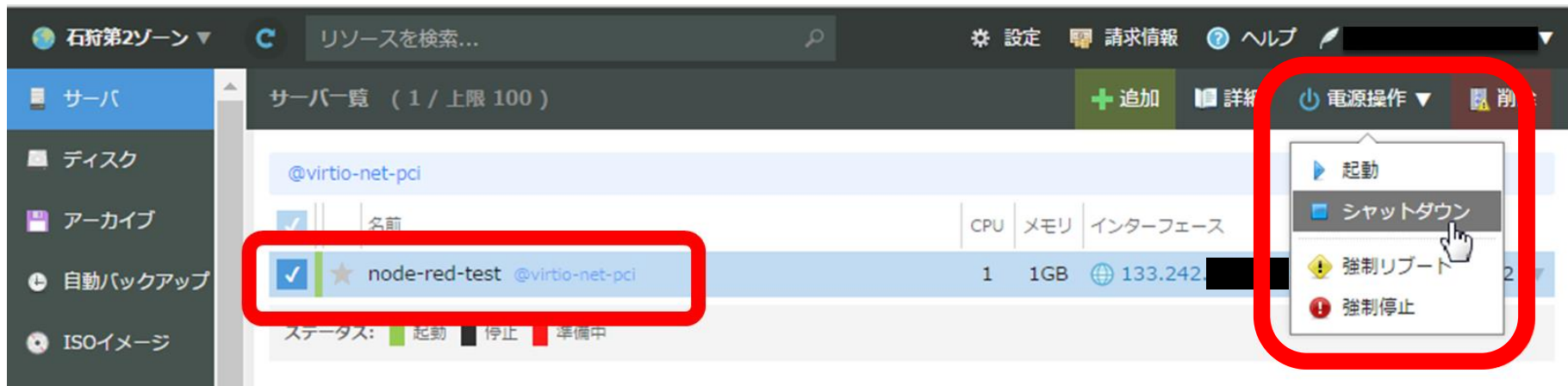
## プロジェクトの削除

ID	2321
名称	New Project

- 削除したプロジェクトは元に戻せません
- 設定済みの連携サービス設定も削除されます



グローバルIPアドレスを持つサーバは攻撃対象になりますので、作成いただいたサーバは削除します。サーバがまだ起動している場合、さくらのクラウドで対象サーバのチェックを入れ、[電源操作]から【シャットダウン】をクリックします。



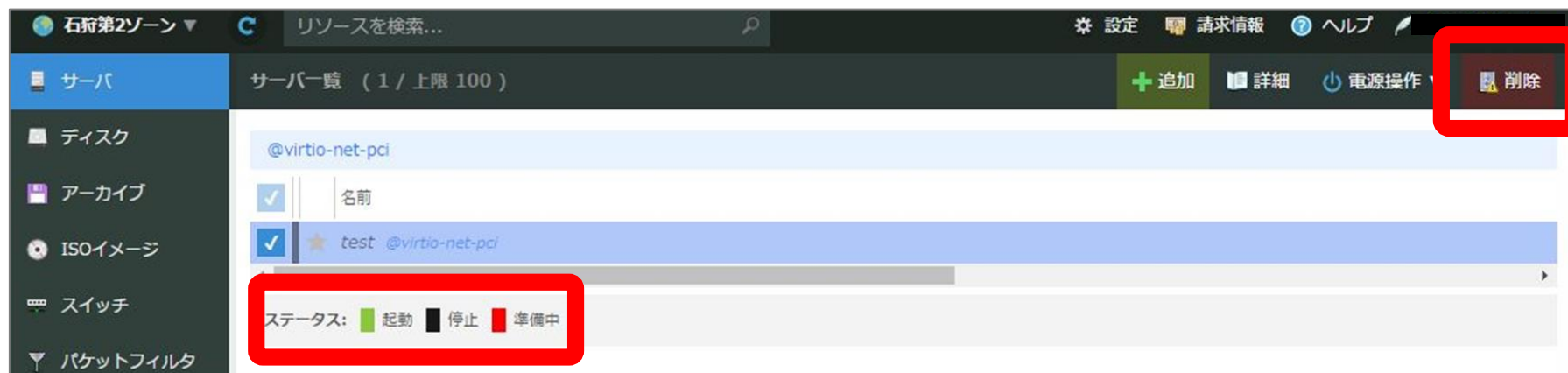
シャットダウン対象を確認のうえ【シャットダウン】をクリックします。  
再度ダイアログにて確認が表示されますので【実行】をクリックします。

The screenshot shows the Sakura Cloud management console. On the left is a navigation menu with items like 'サーバ', 'ディスク', 'アーカイブ', etc. The main area displays a list of resources. A red box highlights the 'シャットダウン' button in the top right of the resource list. Below the list, a modal dialog titled '操作確認' (Operation Confirmation) is open, with a red box highlighting the '実行' (Execute) button. The dialog text reads: 'サーバ シャットダウン' and '一つまたは複数のリソースに対して操作を実行します'.

リソースID	名前	CPU	メモリ	インターフェース
[REDACTED]	★ node-red-test @virtio-net-pci	1	1GB	[REDACTED]



シャットダウンが正常に実行されると、チェックボックス横のラインが「緑→赤→灰」と遷移します。ラインが灰色に変化したら、再度対象となるサーバにチェックを入れ、【削除】をクリックします。



今回は[接続されたディスク]についても削除しますのでチェックを入れ、【削除】をクリックします。  
ダイアログが表示されますので【実行】をクリックします。

The screenshot shows the Sakura Cloud management interface. On the left is a navigation menu with categories like 'サーバ' (Servers), 'ディスク' (Disks), and 'アーカイブ' (Archives). The main area displays a table of resources. A modal dialog titled '操作確認' (Operation Confirmation) is open, asking for confirmation to delete the server. The '実行' (Execute) button in the dialog is highlighted with a red box. In the background, the '削除' (Delete) button in the top right corner is also highlighted with a red box. The table below shows the selected resource:

リソースID	名前	CPU	メモリ	インターフェース
[Redacted]	★ test @virtio-net-pci	1	1GB	[Redacted]

接続されたディスク: サーバの削除後に接続されていたディスクも削除する場合は選択してください

<input checked="" type="checkbox"/>	リソースID	名前
<input checked="" type="checkbox"/>	[Redacted]	test

削除の工程が表示されます。全てのステータスが成功となれば削除は完了です。

The screenshot shows the Sakura Cloud management console. A modal dialog titled "サーバ 削除" (Server Deletion) is open, displaying a table of resources to be deleted. The table has columns for Name, Method, Resource, and Status. The resources listed are a server named "test" and a disk named "test". The server's status is "要求" (Requested) and the disk's status is "待機中" (Waiting). The dialog includes "中断" (Cancel) and "閉じる" (Close) buttons.

名前	メソッド	リソース	ステータス
サーバ: test	DELETE	cloud/1.1/server/ [redacted]	要求
ディスク: test	DELETE	cloud/1.1/disk/ [redacted]	待機中

以上でハンズオンにおける作業は全て終了となります。  
お疲れ様でした！

さくらインターネットでは、「さくらクラブ」としてハンズオン等のイベントをともに開催いただける仲間(部員)を募集しています。  
(テーマはIoTに限らず、クラウドやVPS、スタートアップ系ネタでもOK!)

ご興味があれば、Come and join us & Feel free to contact me!

連絡先 : sakura-club@sakura.ad.jp  
さくらクラブ : <http://www.slideshare.net/MasayaHayashi/lt20151224>